ISSN:0975-9646

Enhancing Data Integrity in Public Clouds

Malathi.P
PG scholar, Muthayammal College Of Engineering,
Rasipuram, Nammakkal(DT), Tamilnadu,
India.

Abstract --- A Cloud is a collection of terminals and servers that are publicly accessible via the internet. One of the primary use of cloud computing is data storage. In Cloud computing data are stored in encrypted form to ensure confidentiality. For more confidentiality two layer encryption approaches is implemented. The third party auditor will audit the data files and stored in cloud environment. There is chance to third party auditor will change the data. My proposed system has the notification method. If Third party auditor attempts to modify the data, the application will sends notification to the corresponding data owner. The third party auditor has rights to audit the data only. Until the user verifies the notification, the modification is not committed to the database.

Key words: user data, encryption, two layer encryption, third party auditor, Notification

I. INTRODUCTION

In cloud storage user data will store into the cloud service provider. Security and privacy represent major concerns in the adoption of cloud technologies for data storage. In cloud computing environment main advantage is using the Third Party Auditor (TPA). That third party auditor will audit the cloud environment. Auditing will performed as two ways. First is CSP side and another is user side. In CSP side auditing is resource access and maintains, scheduling, security. In server side auditing process is space auditing, resource/ Data integrity, Security, storing/ retrieving, Data analysis. In this paper will consider the user side auditing, especially the data integrity. One way for increasing data integrity is encrypting the data and providing Access control policies.

However, whereas encryption assures the confidentiality of the data against the cloud, the use of conventional encryption approaches is not sufficient to support the enforcement of fine-grained organizational access control policies (ACPs). Many organizations have today ACPs regulating which users can access which data; these ACPs are often expressed in terms of the properties of the users, referred to as *identity attributes*, using access control languages such as XACML. Such an approach, referred to as *attribute-based access control* (ABAC), supports fine-grained access control which is crucial for high-assurance data security and privacy. Supporting ABAC over encrypted data is a critical requirement in order to utilize cloud storage services for selective data sharing among different users. Notice that often user identity attributes encode private information and should be strongly protected from the cloud, very much as the data themselves. Approaches based on encryption have been proposed for fine-grained access control over encrypted data[2],[3].As shown in Figure1,those approaches group data items based on ACPs and encrypt each group with a different symmetric key. Users then are given only the keys for the data items they are allowed to access. Extensions to reduce the number of keys that need to be distributed to the users have been proposed exploiting hierarchical and other relationships among data items. Such approaches however have several limitations:

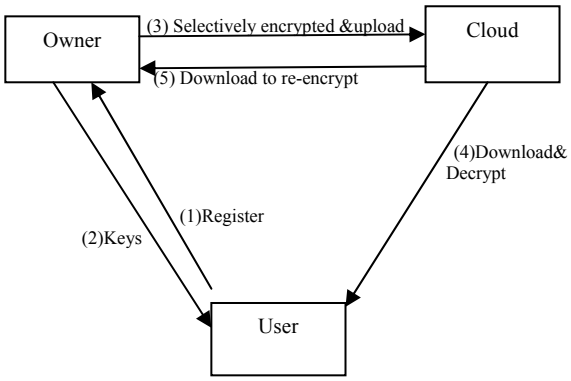

Fig.1: Traditional approach

Recently proposed approaches based on broadcast key management schemes [4], [5], [6] address some of the above limitations. We refer to these approaches as *single layer encryption* (SLE) approaches, since, like previous approaches, they require the data owner to enforce access control through encryption performed at the data owner. However, unlike previous approaches, SLE assures the privacy of the users and supports fine-grained ACPs.

However, while SLE addresses some limitations of previous approaches, it still requires the data owner to enforce *all* the ACPs by fine-grained encryption, both initially and subsequently after users are added/revoked or the ACPs change. All these encryption activities have to be performed at the owner that thus incurs high communication and computation cost. For example, if an ACP changes, the owner must download from the cloud the data covered by this ACP, generate a new encryption key, re-encrypt the downloaded data with the new key, and then upload the re-encrypted data to the cloud. In this paper, we propose a new approach to address this shortcoming. The approach is based on two layers of encryption applied to each data item uploaded to the cloud. Under this approach, referred to as *two layer encryption* (TLE), the data owner performs a coarse grained encryption over the data in order to assure the confidentiality of the data from the cloud. Then the cloud performs fine grained encryption over the encrypted data provided by the data owner based on the ACPs provided by the data owner. It should be noted that the idea of two layer encryption is not new. However, the way we perform coarse and fine grained encryption is novel and provides a better solution than existing solutions based on two layers of encryption[7].We elaborate in

details on the differences between our approach and and existing solutions in the related work section existing solutions in the related work section.

A challenging issue in the TLE approach is how to decompose the ACPs so that fine-grained ABAC enforcement can be delegated to the cloud while at the same time the privacy of the identity attributes of the users and confidentiality of the data are assured. In order to delegate as much access control enforcement as possible to the cloud, one needs to decompose the ACPs such that the data owner manages minimum number of attribute conditions in those ACPs that assures the confidentiality of data from the cloud. Each ACP should be decomposed to two sub ACPs such that the conjunction of the two sub ACPs result in the original ACP. The two layer encryption should be performed such that the data owner first encrypts the data based on one set of sub ACPs and the cloud re-encrypts the encrypted data using the other set of ACPs. The two encryptions together enforce the ACP as users should perform two decryptions to access the data. For example, if the ACP is (C1 ∧ C2) ∨ (C1∧C3), the ACP can be decomposed as two sub ACPs C1 and C2 ∨ C3. Notice that the decomposition is consistent; that is, (C1 ∧ C2) ∨ (C1 ∧ C3) = C1 ∧ (C2 ∨ C3). The data owner enforces the former by encrypting the data for the users satisfying the former and the cloud enforces the latter by re-encrypting the data owner encrypted data for the users satisfying the latter. Since the cloud does not handle C1, it cannot decrypt owner encrypted data and thus confidentiality is preserved. Notice that users should satisfy the original ACP to access the data by performing two decryptions. In this paper, we show that the problem of decomposing ACPs such that the data owner manages the minimum number of attribute conditions while at the same time assuring the confidentiality of the data in the cloud is NP-complete. We propose two optimization algorithms to find the near optimal set of attribute conditions and decompose each ACP into two sub ACPs.

The TLE approach has many advantages. When the policy or user dynamics changes, only the outer layer of the encryption needs to be updated. Since the outer layer encryption is performed at the cloud, no data transmission is required between the data owner and the cloud. Further, both the data owner and the cloud service utilize a broadcast key management scheme [8] whereby the actual keys do not need to be distributed to the users. Instead, users are given one or more secrets which allow them to derive the actual symmetric keys for decrypting the data.

Then the auditing will performed by the user using the Third Party auditor (TPA). The user will give the kinds about the data for auditing the user data. So the third party auditor can see the encrypted data and audit the data. There is change to third party auditor attempts to modify the content of the data. that is major disadvantage of using third party auditor for auditing. To over come this disadvantage in this paper will implement the notification method.

In that if a third party auditor attempts to modify the content means the notification will generate and the sends to the user. Until the user verification the modification will not commit to the cloud service provider.

2 BUILDING BLOCKS

In this section we first introduce broadcast encryption schemes [9], [10] and oblivious commitment based envelope protocols [11]. We present an abstract view of the main algorithms of those protocols and then describe how we use them to build our privacy-preserving attribute based group key management (PPAB-GKM) scheme [8]. We then present an overview of the SLE approach [4], [5], [6] which is used as the base model for comparison with the TLE approach proposed in this paper.

A. Single Layer Encryption Approach

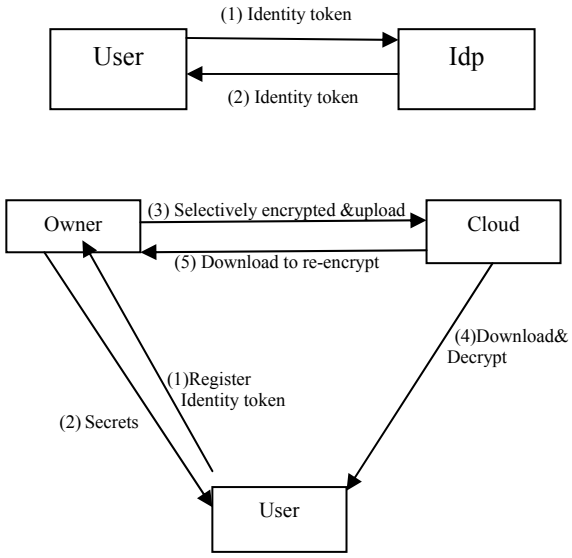

Fig.2: Single Layer Encryption approach

As shown in Figure 3, the SLE approach follows the conventional data outsourcing scenario where the Owner enforces all ACPs through selective encryption and uploads encrypted data to the untrusted Cloud. The system goes through five different phases. We give an overview of the five phases below:

Identity token issuance: IdPs issue identity tokens to Usrs based on their identity attributes.

Identity token registration: Usrs register all their identity tokens to obtain secrets in order to later decrypt the data that they are allowed to access. **Data encryption and uploading:** Based on the secrets issued and the ACPs, the Owner encrypts the data using the keys generated using the AB-GKM: KeyGen algorithm and uploads to the Cloud.

Data downloading and decryption: Usrs download encrypted data from the Cloud and decrypt using the key derived from the AB-GKM: KeyDer algorithm. **Encryption evolution management:** Over time, either access control polices or user credentials may change. Further, already encrypted data may go through frequent updates. In such situations, it may be required to re-encrypt already encrypted data. The Owner alone is responsible to perform such re-encryptions. The Owner downloads all

affected data from the Cloud, decrypts them and then follows the data encryption and upload step.

B. Two Layer Encryption

Figure3 shows the system diagram of the TLE approach. The system goes through one additional phase compared to the SLE approach. We give an overview of the six phases below:

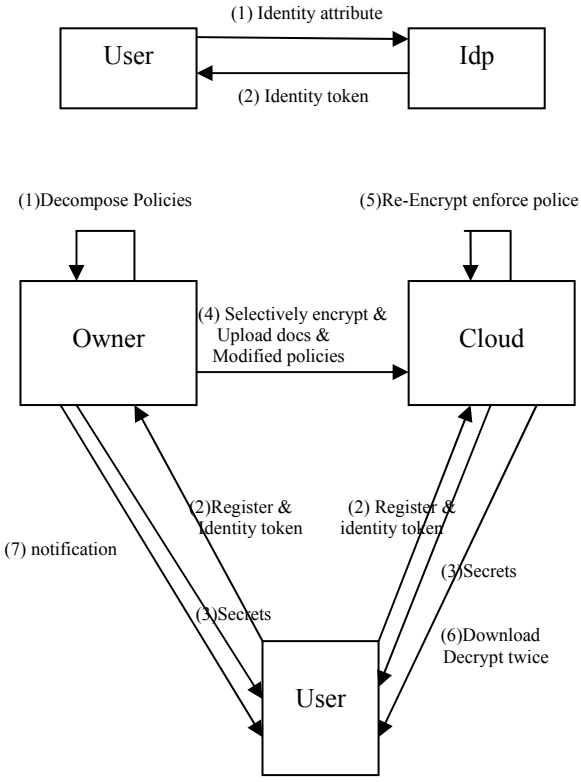

Fig.3: Two Layer Encryption approach

Identity token issuance: IdPs issue identity tokens to Usrs based on their identity attributes

Policy decomposition: The Owner decomposes each ACP into at most two sub ACPs such that the Owner enforces the minimum number of attributes to assure confidentiality of data from the Cloud. It is important to make sure that the decomposed ACPs are consistent so that the sub ACPs together enforce the original ACPs. The Owner enforces the confidentiality related sub ACPs and the Cloud enforces the remaining sub ACPs.

Identity token registration: Usrs register their identity tokens in order to obtain secrets to decrypt the data that they are allowed to access. Usrs register only those identity tokens related to the Owner's sub ACPs and register the remaining identity tokens with the Cloud in a privacy preserving manner. It should be noted that the Cloud does not learn the identity attributes of Usrs during this phase.

Data encryption and uploading: The Owner first encrypts the data based on the Owner's sub ACPs in order to hide the content from the Cloud and then uploads them along with the public information generated by the AB-GKM::KeyGen algorithm and the remaining sub ACPs to the Cloud. The Cloud in turn encrypts the data based on the keys generated using its own AB-GKM::KeyGen algorithm. Note that the AB-GKM::KeyGen at the Cloud takes the secrets issued to Usrs and the sub ACPs given by the Owner into consideration to generate keys.

Data downloading and decryption: Usrs download encrypted data from the Cloud and decrypt the data using the derived keys. Usrs decrypt twice to first remove the encryption layer added by the Cloud and then by the Owner. As access control is enforced through encryption, Usrs can decrypt only those data for which they have valid secrets.

Encryption evolution management: Over time, either ACPs or user credentials may change. Further, already encrypted data may go through frequent updates. In such situations, data already encrypted must be re-encrypted with a new key. As the Cloud performs the access control enforcing encryption, it simply re-encrypts the affected data without the intervention of the Owner.

3. POLICY DECOMPOSITION

Recall that in the SLE approach, the Owner incurs a high communication and computation overhead since it has to manage all the authorizations when user dynamics or ACPs change. If the access control related encryption is some how delegated to the Cloud, the Owner can be freed from the responsibility of managing authorizations through re-encryption and the overall performance would thus improve. Since the Cloud is not trusted for the confidentiality of the outsourced data, the Owner has to initially encrypt the data and upload the encrypted data to the cloud. Therefore, in order for the Cloud to allow to enforce authorization policies through encryption and avoid re-encryption by the Owner, the data may have to be encrypted again to have two encryption layers. We call the two encryption layers as *inner encryption layer* (IEL) and *outer encryption later* (OEL). IEL assures the confidentiality of the data with respect to the Cloud and is generated by the Owner. The OEL is for fine-grained authorization for controlling accesses to the data by the users and is generated by the Cloud. An important issue in the TLE approach is how to distribute the encryptions between the Owner and the Cloud. There are two possible extremes. The first approach is for the Owner to encrypt all data items using a single symmetric key and let the Cloud perform the complete access control related encryption. The second approach is for the Owner and the Cloud toper form the complete access control related encryption twice. The first approach has the least overhead for the Owner, but it has the highest information exposure risk due to collusions between Usrs and the Cloud. Further, IEL updates require re-encrypting all data items. The second approach has the least information exposure risk due to collusions, but it has the highest overhead on the Owner as the Owner has to perform the same task initially as in the SLE approach and, further, needs to manage all identity attributes. An alternative solution is based on decomposing ACPs so that the information exposure risk and key management overhead are balanced. The problem is then how to decompose the ACPs such that the Owner has to manage the minimum number 7 of attributes while

delegating as much access control enforcement as possible to the Cloud without allowing it to decrypt the data. In what follow we propose such an approach to decompose and we also show that the policy decomposition problem is hard.

3.1 Policy Cover

We define the *policy cover problem* as the optimization problem of finding the minimum number of attribute conditions that "covers" *all* the ACPs in the ACPB. We say that a set of attribute conditions covers the ACPB if in order to satisfy *any* ACP in the ACPB, it is necessary that at least one of the attribute conditions in the set is satisfied. We call such a set of attribute conditions as the *attribute condition cover*. For example, if ACPB consists of the three simple ACPs {C1 ∧ C2, C2 ∧ C3, C4}, the minimum set of attributes that covers ACPB is {C2, C4}. C2 should be satisfied in order to satisfy the ACPs C1 ∧ C2 and C2 ∧ C3. Notice that satisfying C2 is not sufficient to satisfy the ACPs. The set is minimum since the set obtained by removing either C2 or C4 does not satisfy the cover relationship. We define the related decision problem as follows.

Definition 6(POLICY-COVER): Determine whether ACPB has a cover of k attribute conditions.

The following theorem states that this problem is NP-complete.

Theorem 1: The POLICY-COVER problem is NP-complete.

Proof: We first show that POLICY-COVER ∈ NP. Suppose that we are given a set of ACPs ACPB which contains the attribute condition set AC, and integer k. For simplicity, we assume that each ACP is a conjunction of attribute conditions. However, the proof can be trivially extended to ACPs having any monotonic Boolean expression over attribute conditions. The certificate we choose has a cover of attribute conditions AC ⊂AC. The verification algorithm affirms that |AC | = k, and then it checks, for each policy in the ACPB, that atleast one attribute condition in AC is in the policy. This verification can be performed trivially in polynomial time. Hence, POLICY-DECOM is NP. Now we prove that the POLICY-COVER problem is NP-hard by showing that the vertex cover problem, which is NP-Complete, is polynomial time reducible to the POLICY-COVER problem. Given an undirected graph G =(V,E)and an integer k, we construct a set of ACPs ACPB that has a cover set of size k if and only if G has a vertex cover of size k. Suppose G has a vertex cover V⊂ V with |V | = k. We construct a set of ACPs ACPB that has a cover of k attribute conditions as follows. For each vertex vi ∈ V, we assign an attribute condition Ci.

Since thePOLICY-COVER problem is NP-complete, one cannot find a polynomial time algorithm for finding the minimum attribute condition cover. In the following section we present two approximation algorithms for the problem.

Algorithm 1 GEN-GRAPH

```
1: C = φ
2: for Each ACPi ∈ACPB, i =1 to Np do
3: ACP ← Convert ACPi to DNF
4: for Each conjunctive term c of ACP do
5: Add c to C
6: end for
7: end for
8: //Represent the conditions as a graph
9: G =(E,V), E = φ, V = φ
10: for Each conjunctive term ci ∧ C, i =1 to Nc do
11: Create vertex v, if v ∈ V, for each AC in ci
12: Add an edge ei between vi and each vertex already added for ci
13: end for
14: Return G
```

We give a high-level overview of the GEN-GRAPH algorithm 1. It takes the ACPB as the input and converts each ACP into DNF(disjunctive normal form). The unique conjunctive terms are added to the set C. For each attribute condition in each conjunctive term in C, it creates a new vertex in G and adds edges between the vertices corresponding to the same conjunctive term. Depending on the ACPs, the algorithm may create a graph G with multiple disconnected sub graphs.

Algorithm 2 APPROX-POLICY-COVER1

```
1: G = GEN-GRAPH (ACPB)
2: ACC = φ
3: for Each disconnected subgraph Gi =(Vi,Ei) of G do
4: if |Vi| == 1 then
5: Add ACi corresponding to the vertex to ACC
6: else
7: while Ei 6= φ do
8: Select a random edge (u,v)of Ei
9: Add the attribute conditions ACu and ACv corresponding to {u,v} to ACC.
10: Remove from Ei every edge incident on either u or v
11: end while
12: end if
13: end for
14. Return ACC
```

3.2Policy Decomposition

The Owner manages only those attribute conditionsin ACC.The Cloud handles the remaining set of attribute conditions, ACB/ACC. The Owner re-writes its ACPs such that they cover ACC. In other words, the Owner enforces the parts of the ACPs related to the ACs in ACC and Cloud enforces the remaining parts of the policies along with some ACs in ACC. The POLICYDECOMPOSITION algorithm 3 shows how the ACPs are decomposed into two sub ACPs based on the attribute conditions in ACC.

Algorithm 3 POLICY-DECOMPOSITION

1: ACPBOwner = φ
2: ACPBCloud = φ
3: **for** Each ACPi in ACPB **do**
4: Convert ACPi to DNF
5: ACPi(owner)= φ
6: ACPi(cloud)= φ
7: **if** Only one conjunctive term **then**
8: Decompose the conjunctive term c into c1 and c2 such that ACs in c1 ∈ ACC, ACs in c2 ∈ ACC and c = c1 ∧ c2
9: ACPi(owner)= c1
10: ACPi(cloud)= c2
11: **else if** At most one term has more than one AC **then**
12: **for** Each single AC term c of ACP′ do i
13: ACPi(owner)∨= c
14: ACPi(cloud)∨= c
15: **end for**
16: Decompose the multi ACterm c intoc1 and c2 such that ACs in c1 ∈ ACC, ACs in c2 ∈ ACC and c = c1 ∧ c2
17: ACPi(owner)∨= c1
18: ACPi(cloud)∨= c2
19: **else**
20: **for** Each conjunctive term c of ACP′ **do** i
21: Decompose c into c1 and c2 such that ACs in c1 ∈ ACC, ACs in c2 ∈ ACC and c = c1 ∧ c2
22: ACPi(owner)∨= c1
23: **end for**
24: ACPi (cloud)= ACP′ i
25: **end if**
26: Add ACPi(owner) to ACPBOwner
27: Add ACPi(cloud) to ACPBCloud
28: **end for**
29: Return ACPBOwner and ACPBCloud

Algorithm 3 takes the ACPB and ACC as input and produces the two sets of ACPs ACPB Owner and ACPBCloud that are to be enforced at the Owner and the Cloud respectively. It first converts each policy into DNF and decompose each conjunctive term into two conjunctive terms such that one conjunctive term has only those AC sin ACC and the other term may or may not have the ACs in ACC. It can be easily shown that the policy decomposition is consistent. That is, the conjunction of corresponding sub AC Psin ACPB Owner and ACPBCloud respectively produces an original ACP in ACPB.

7. NOTIFICATION

In cloud enivronment third party auditor is auditing the data content and stored in cloud storage.

There is chace to third party auditor will attempt to modify the contents. There is user have not knowledge about the third party auditor modification. So in this paper will implements the notification method. If a third party auditor attempts to modify the content menas the notification sends to the user. The notification will generated by the notification application which is stored in the cloud service provider. In main advatage of this application untill the user verification the modification will not commits to the cloud service provider. Then that have time slot also. Untill the paticular time the cloud service provider did't receive any acknowledgement frome the user means the modification will commit to the RollBack.

In this way the third party auditor modification will not stored in the cloud space. So the integrity of the data is incresed.

8 .CONCLUSIONS

Current approaches to enforce ACPs on outsourced data using selective encryption require organizations to manage all keys and encryptions and upload the encrypted data to the remote storage. Such approaches incur high communication and computation cost to manage keys and encryptions whenever user credentials or organizational authorization policies/data change. Then if attacker attack the Cloud Service Provider means That Encrption File Will Motified or attack the Original Information.So the Security Policies Will affected.

In this Paper we proposed a two layer encryption based approach to solve this problem by delegating as much of the access control enforcement responsibilities as possible to the Cloud while minimizing the information exposure risksdue to colluding Usrs and Cloud.Akeyproblem in this regard is how to decompose ACPs so that the Owner has to handle a minimum number of attribute conditions while hiding the content from the Cloud. We showed that the policy decomposition problem is NP-Complete and provided approximation algorithms. Based on the decomposed ACPs, we proposed a novel approach to privacy preserving finegrained delegated access control to data in public clouds.Our approach is based on a privacy preserving attribute based key management scheme that protects the privacy of users while enforcing attribute based ACPs.As the experimental results show,decomposing the ACPs and utilizing the two layer of encryption reduce the over head at the Owner. Then The Security of Cloud Service Provider Will increased.So the confidentialityof the data and encrpted file will be high. In that if a third party auditor will attempts to modify the content means the client received the notification. Untill the user verification the modification will not commits to the cloud storage. In that way we can reduce the others modifing content then confidentiality also increased.

REFERENCES

[1] Alina Oprea, Michael K. Reiter,Ke Yang 'space-efficient block storage integrity' In Proc of NDSS on 2005.

[2] C. C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia 'An improved dynamic provable data possession model' Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on Sept. 2011.

[3] Cong Wang, Qian Wang, and Kui Ren 'Ensuring data storage security in Cloud Computing' *Quality of Service IWQoS 17th International Workshop on July 2009.*

[4] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik 'Scalable and efficient provable data possession' *Secure Comm* Security and Privacy in Communication Networks, on June 2013, Volume 18, Issue 3, pp 265-271.

[5] Hovav Shacham, Brent Waters 'compact proofs of retrievability' Journal of cryptology july 2013, Volume 26, Issue 3, pp 442-483.

[6] Junkil Park,Jin-Young choi, 'Formal Security Policy Model For Commaon Criteria Evaluation' Advanced Communication Technology, the 9th international Conference (Volume:1) on Feb 2007.

[7] Mehul A. Shah Ram Swaminathan Mary Baker 'privacy-preserving audit and extraction of digital contents' Hewlett-Packard, on Apr 2008.

[8] Mohamed Nabeel, Ellisa Bertino ' Privacy preserving delegated access control in public clouds' IEEE International Conference on 2013.

[9] Qian Wang, Kui Ren ,Wenjing Lou ,Yanchao Zhang 'Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance' INFOCOM 2009, IEEE on April 2009.