HAL
archives-ouvertes.fr

Protecting outsourced data in cloud computing: novel politic of security based on capability list

Khalid Aissaoui, Hicham Belhadaoui, Abdelouahed Zakari, Mounir Rifi

▶ **To cite this version:**

Khalid Aissaoui, Hicham Belhadaoui, Abdelouahed Zakari, Mounir Rifi. Protecting outsourced data in cloud computing: novel politic of security based on capability list. Revue Méditerranéenne des Télécommunications=Mediterranean Telecommunication Journal, CNRST-IMIST, 2016, 6 (2). hal-03235613

HAL Id: hal-03235613
https://hal.univ-lorraine.fr/hal-03235613

Submitted on 20 Aug 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

PROTECTING OUTSOURCED DATA IN CLOUD COMPUTING: NOVEL POLITIC OF SECURITY BASED ON CAPABILITY LIST

Khalid Aissaoui[*1], Hicham Belhadaoui*[2], Abdelouahed Zakari[#3], Mounir Rifi[*4]

*RITM Laboratory, CED Engineering Sciences Ecole Supérieure de Technologie, Hassan II University of Casablanca, Morocco
[1]aissaoui.khalid@gmail.com
[2]belhadaoui_hicham@yahoo.fr
[4]rifi@email.com
#LITA University of LORRAINE Metz, France
[3]abdelouahed.zakari@univ-lorraine.fr

Abstract:
Cloud computing is a new concept that was developed quite a lot these recent years. The number of organizations and companies adopting the cloud solutions is increasing because of many advantages found in its use (cut in setup and operating costs, scalability, flexibility….). However, this new paradigm faces some concerns that slow down its evolution and cause less infatuation from the users. The main issue here is undoubtedly the lack of security and mainly how outsourced sensible data within a cloud server can be protected?
A number of solutions are being examined in order to strengthen the used models (authentication, data encryption and fragmentation before storing, Identity Access Management…), Data owners, users and Cloud Service providers are often involved. In this paper, we propose a new approach in which the provider does not participate on any level in the access management, only the owner can setup, update and verify access rights with the aim of minimizing leak risks.

Keywords: Cloud computing security, cryptography, confidentiality, access management.

1 Introduction:

1.1 What is Cloud Computing?

During the last few years, new technologies of information have made considerable progress; internet became largely used in all domains, offering larger bandwidth and bigger storage capacity. Consequently, a new IS management model was advanced.
Cloud Computing is a concept which allows the provisioning of scalable, virtualized, distributed and multiple resources, hardware (Networks, servers, storing…) and/or software ones (Applications, services, platforms, OS…) over a simple internet connection by vendors or providers.
Cloud Computing offers many great advantages. First, the same block of data can be stored separately in more than one server; this provides best availability such as Back up. Besides, one pays only for the resources that he actually uses (pay as you go) without need for more expenses (purchasing new devices, hiring more engineers, computer maintenance…). Finally, data stored on Cloud provider's servers can be reached from anywhere using a simple internet connection.

1.2 Cloud's deployment models and services:

There are three different models that can be deployed on provider's servers [1]. **Public** Cloud is intended for everyone. Users need a simple internet connection and in this case data security is a minor issue. In the **private** model, providers offer more secure solutions (applications, software, hardware…) to organizations or companies. Finally, **hybrid** Cloud is a combination of private and public deployment.
According to Jansen and Grance [2], Cloud providers offer diversified services depending on users or client requirements. **Software as a Service (SaaS);** Users get access to applications and software using a web browser. Saas enables providers to control and manage the use of their products. **Platform as a Service (PaaS);** In this model, providers offer a development environment (IDE, toolkits, DBMS…) to developers in order to design, develop and deploy their own solutions. **Infrastructure as a Service (IaaS);** IaaS enables organizations and companies to exclusively use hardware (Servers, routers, connectors…).
We aim to describe a novel approach to protect and secure data in Cloud environment; hence we exclude providers from participating in the access management. Only the data owner can manage and update the access rights.
To explain our concept, this paper is divided as follows: First, we enumerate security problems that face Cloud Computing. Second, we describe some proposed solutions to secure outsourced data. Third, we present an implementation of our concept, its features and advantages. Finally, we conclude with some perspectives of evolution.

2. Security in Cloud:

Cloud Computing is being used increasingly due to its significant advantages. However, there are some concerns about data security that require more attention and need to be resolved. Within this new paradigm, traditional means of protection are deprecated. Sensitive data like requirements, financial information or scientific formula are moved to shared servers managed by a third party which must guarantee security as agreed with clients (data owner or users).

In a cloud environment, sensitive data is stored on shared remote servers assigned by a service provider who has to guarantee confidentiality. Access to this data is forbidden to persons or groups not allowed including the provider. Some techniques are used, like cryptography (data is encrypted before being sent to external servers), or fragmentation which is an interesting alternative too (columns of the relational data table, which together are more sensitive, are stored separately on different nodes).

Besides, authentication is an important step which allows the server to identify any user in order to enable access to data. Different methods can be implemented to strengthen this process (passwords, strong authentication and other challenge-response authentication).

Furthermore, to guarantee data integrity, the owner has to protect his/her files from modification or deletion that may occur (accidentally or deliberately).

Users who need access to stored data in cloud environment have different profiles and roles; they are employees, customers or even suppliers. Hence, the system needs an identity access management to give the right data to the right user (confidentiality).

Finally, the non-repudiation service provides proof of the integrity and origin of the data. This prevents the user from denying actions performed (purchasing on the net, sending messages…).

3. Related work:

Protecting and managing access control to outsourced data has been the main issue in some research papers. Naor and al [3] use a mechanism of generation and distribution of symmetric keys based on Blom's work [4]. In their paper, Ateniese and al [5] propose a proxy re-encryption; the owner encrypts with symmetric content keys blocks of data before sending it to cloud servers, those content keys are encrypted with the owner's master public key. His master private key and users public keys are then combined to generate proxy re-encryption keys which are used to recover plain text intended to a specific user. In this model, any collaboration between the services provider and the user would compromise data security by revealing decryption keys.

In [6], Yu and al propose an encryption based on attributes assigned to data. Every file is associated to an attribute. The access structure of each user (defining his authorized files) is created according to a logical expression over these attributes. Each file is encrypted with the public key corresponding to his attribute. The user is not able to decrypt a file if its attribute does not match his access structure. In this scheme, the computation of logical expressions becomes more complex as the amount of data stored grows.

Miklau and al [7] are interested in access control on XML files by encrypting different portions of an XML tree with different keys and using metadata nodes.

For securing their outsourced data [8], Vermacati and al propose a solution based on a key derivation system. Files are encrypted with symmetric keys; a private key is assigned to each file. A user can only decrypt the files he is allowed to, by using both his private key and a set of public tokens that the owner generates and sends to the server which is responsible for distribution. In this model the complexity of all these operations is linear to the number of users.

Hota and al propose a solution [9] in which the data owner shares with the provider a list of access rights (the capability list). It is constantly updated in case of modification or insertion. The provider uses the capability list to manage access requests. Data is encrypted before being sent to the servers with a symmetric key (shared with appropriate users). This approach has some security issues revealed by Gao et al in their paper [10], particularly, *Repeat* and *Man In The Middle* attacks.

The Cloud Service Provider (CSP) participates directly in the access control which requires an additional encryption layer to secure sensitive data.

In the next section, we propose a new scheme using the capability list mentioned above, but the CSP is not involved in securing data and access control.

4. The proposed approach:

4.1 Hota and al's model:

First, we explain here Hota's model on which we base our solution. There are three actors; Data owner DO, Cloud Service Provider CSP and User

Figure 1 shows the architecture used.

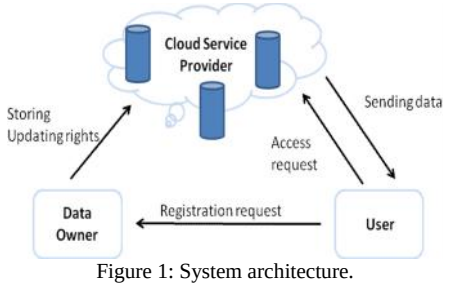

Figure 1: System architecture.

The model is based on three main algorithms. First, DO stores encrypted data and the capability list (which describes users's access rights) on CSP servers. The second algorithm explains how User sends a registration request. Finally, CSP provides on demand and after checking the capability list encrypted data to appropriate users. To achieve their solution they assume that public keys are available for all

participants, without using PKI. Besides, DO does not have to be permanently connected.

Notation	Description
PUSP	CSP public key
PRSP	CSP private key
PUUSR	User public key
PRUSR	User private key
PUOWN	DO public key
PROWN	DO private key
Fi	ith File
Di	ith file message digest
Oi	ith Object
EOi	Ith Encrypted Objet
EK	Encryption
DK	Decryption
Ko	DO symmetric key
MD5	Hash Algorithm
CapList	Capability List (UID, FID, AR)
AR	Access Right (0 r, 1 w, 2 r/w)
UID	User Identity
FID	File Identity

Table 1: shows the notations used in Hota's scheme.

First, DO sends data in encrypted form with the Capability List (UID, FID and AR) (*Figure 2 to 5*).

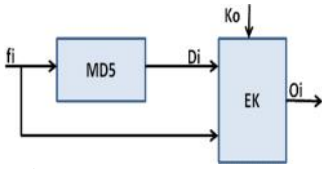

Figure 2: The ith file and its message digest (generated with MD5) are encrypted with DO symmetric key.

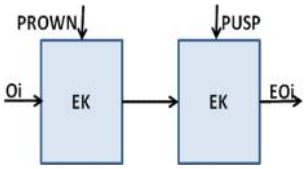

Figure 3: Successive encryption with DO key and CSP public key to establish authentication and confidentiality.

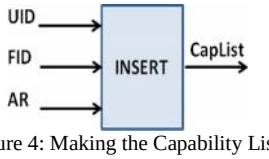

Figure 4: Making the Capability List.

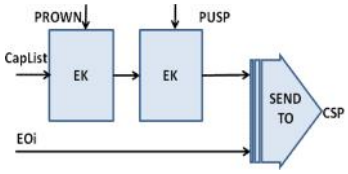

Figure 5: Securing and sending the CapList to CSP.

On the other hand, User needs to send a registration request to DO containing UID, FID and the access rights AR (*figure 6*). DO updates the CapList and sends it with all decryption parameters to CSP (those parameters are necessarily encrypted with User public key) as shown in *figure 7.a*.

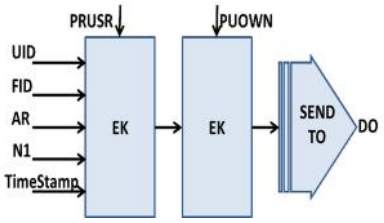

Figure 6: Sending a registration request to DO

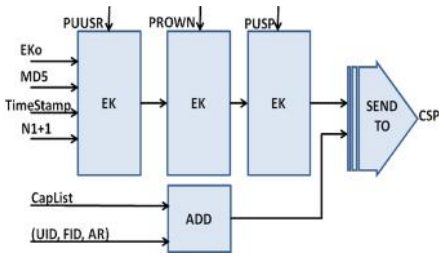

Figure 7.a: Updating and sending the CapList to CSP.

CSP updates CapList stored on his/her servers and sends decryption parameters to User (*figure 7.b*).

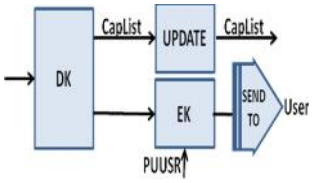

Figure 7.b: Sending the decryption parameters to the User

CSP receives both encrypted data and Capability List and stores it on servers. To get access to stored data, User sends request to CSP who verifies User's identity and access rights before authorizing. This exchange is secured by using session key shared with the Diffie-Hellman algorithm.

4.2 The revised model:

As described above in the scheme proposed in [9], the CSP participates actively in the data access control, since he verifies CapList and sends data to users. This requires an additional encryption layer (public and private key, session key to secure communication with users). Furthermore, the capability list, which is shared with servers in plain form, represents a risky source of information (requests flow, number of users…). Finally, a malicious collaboration between CSP and any user (e.g. key disclosure) would bring a considerable damage to the system security.

In this paper, we propose to remove CSP from the architecture, (*figure 8*) and avoid any exchange between provider and users.

The new approach is based on the development and integration of two programs (admin and access) on servers in order to address CapList management and respond to access requests.

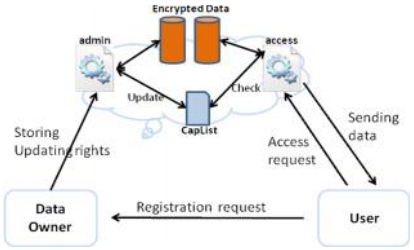

Figure 8: The novel architecture

We assume that all public keys used are available to participants, PKI is not the concern. Besides, DO and users are enabled to execute programs on remote servers provided by CSP.

Only DO has the right to run ***admin*** module, with the aim of updating the capability list and storing encrypted data.

Sending CapList and encrypted data:

Step 1: DO generates the message digest and encrypts it along with the file (fi, MD5(fi)) using a symmetric key Ko.

Step 2: DO needs, first, to log on (authentication) then execute the module ***admin*** to upload encrypted data and capability list. Before that, DO adds his signature to the whole package (Data+CapList) with his private key (PROWN).

Registration request algorithm: manages differently the communication between servers and users.

Step 1: User sends registration request to DO including {UID, FID, Nonce, TimeStamp, AR}. This package is encrypted with PRUSR (signature) then PUOWN (confidentiality).

Step 2 : DO updates the CapList, logs on to server in order to execute ***admin*** module that uploads the new version along with a set of parameters {Ko, MD5, Nonce+1, TimeStamp} encrypted successively with PUUSR and PROWN and sent to concerned User.

Access request: The User executes ***access*** module to get access to data. To do so, User needs, after a valid authentication, to be allowed to run ***access.***

Step 1: User logs on and runs ***access*** with a set of parameters {UID, FID, AR}.

Step 2: The program checks CapList, encrypts with PUUSR (confidentiality) before sending encrypted data.

4.3 Evaluation:

Three aspects of the data security are guaranteed.

Data *integrity*; MD5 is used to calculate message digest by DO. User does the same and compares.

The use of asymmetric cryptography is beneficial in two ways; *confidentiality* is being secured with public key and *non-repudiation* with private key.

Besides, authentication is an important step which strengthens security. It is essential that the system avoids any intrusion to servers by an unauthorized entity, even if data is already encrypted. A strong authentication would be a good answer to this issue.

In conclusion, the threat inferred by the involvement of cloud service provider in the first model is moved away. He is not involved in any part of the scheme.

5. Conclusion:

In this paper, we give an overview of the literature in relation with protecting outsourced data in a cloud environment, and then we propose a novel approach with the aim of improving the system security. It seems that this revised model has a number of advantages as cited above. Therefore, the model is worthy of implementation for further analysis.

Consequently, we aim to set up this architecture in order to analyze results in a practical manner. Later, we plan to refine the model (strong authentication, smart active data).

REFERENCES:

[1] T Mather, S Kumaraswamy, S Latif, Cloud Security and privacy, O'REILLY, September 2009

[2] W Jansen, T Grance, Guidelines on Security and Privacy in Public Cloud Computing, NIST (National Institut of Standards and Technology) U.S. Department of Commerce, December 2011

[3] Dalit Naor, A. Shenhav and A. Wool, Toward Securing Untrusted Storage Without Public-key Operations, Proc. 2005 ACM Workshop on Storage Security and Survivability (StorageSS), Virginia, USA, Nov 2005, pp. 51-56.G.

[4] R. Blom, An Optimal Class of Symmetric Key Generation Systems, Proc. EUROCRYPT 84 Workshop on Advances Cryptology: Theory and Application of Cryptographic Techniques, Springer Verlag, NY, USA, 1985, pp. 335-338.

[5] G. Ateniese, K. Fu, M. Green and S. Hohenberger, Improved Proxy Re-Encryption Schemes with Application to Secure Distributed Storage, ACM Transactions on Information and System Security, Vol. 9 No. 1, Feb 2006 pp. 1-30.

[6] Shucheng Yu, Cong Wang, Kui Ren and Wenjing Lou, Achieving Secure, Scalable and Fine-grained Data Access Control in Cloud Computing, Proc. ACM Workshop on Computer Security Architecture (CSAW'07), Nov 2007, USA.

[7] Miklau and D. Suciu, Contolling Access to Published Data Using Cryptography, Proc. 29th VLDB, Germany, Sept 2003, pp. 898-909.C.

[8] S. D. C. di Vimercati, S; Foresti, S Jajodia, S Parabocshi and P. Samarati, Over-encryption : Management of Access Control Evolution on Outsourced Data, Proc. 33th International Conference on Very Large Databases (VLDB'07), Vienna, Austria, 2007, pp. 123-134.X.

[9] Hota, S. Sanka, M. Rajarajan, S. K. Nair, Capability-Based Cryptographic Data Access Control in Cloud Computing. Int J. Advanced Networking and Applications 03, 1152-1161 (2011)

[10] Gao, Z. Jiang, R. Jiang, A Novel Data Access Scheme in Cloud Computing, International Conference on Computer and Information Application (ICCIA 2012)