Secure Cloud-based Access Control Optimization (SCACO)

P. Visalakshi*, Devashish Sood and S. Priya

Department of Computer Science and Engineering SRM University, Kattankulathur campus, Kancheepuram District - 603 203, Tamil Nadu, India; visalakshi.p@ktr.srmuniv.ac.in, visasan2003@gmail.com, devasood@gmail.com, priya.sn@ktr.srmuniv.ac.in

Abstract

Objective: We propose a decentralized access controlled scheme for secure data storage in clouds that supports public key encryption. Our scheme also has the added feature of generating attribute key-value pairs with ciphers for easily granting permission to transfer data. The scheme prevents tampering by performing integrity checks and verifying the message digest. Methods/Statistical Analysis: In the proposed scheme, the data is transferred by public key encryption among users by the RSA algorithm. The attributes are sent via e-mail and can trigger an encrypted transfer of data stream, which can be decrypted only by the validated users. The cloud does not store the private keys of users and only stores public keys. Findings: The computation and transfer of cipher text, and storage overheads are the same as a centralized scheme. The decentralization of data makes it private and the encrypted copy on the cloud provides a redundant backup which cannot be read by the cloud. The performance of the scheme has been measured by time complexity of the operations using big-O notation. Applications/Improvement: The scheme can be improved with signatures. Signatures can be studied in detail and come in various types. Some signatures verify the authenticity of files while other complex ones work with multi-party authentication.

Keywords: Access Control in Cloud, Cryptography, MVC, RSA

1. Introduction

Cloud computing is gaining traction in our everyday lives as it frees our data from the devices and stores it virtually. The cloud consists of data from various users which is available on the internet with the authentication from user devices. This allows data to be synced and freed from the device failures like the breakdown of hardware or theft. It provides the services such as Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). Platform as a Service provides platforms to developers to create applications. Eg:- Heroku, Google App Engine, and Red Hat’s OpenShift. Software as a Service provides services to non-developers for easy to use software. Eg:- Google Apps, and Dropbox. Infrastructure as a Service provides the ground up access to developers to build things from scratch with the infrastructure. Eg:- Navisite, Exoscale, and Softlayer.

The cloud allows us to share our data online making it available irrespective of geographical location. It also allows us to share data with other users of the cloud. We have developed Secure Cloud-based Access Control and Optimization System to authenticate users to ensure that they are members of the cloud and authorize some of them to access specific data. Access Control in the cloud is a really sensitive area with regards to Isolation of data from unauthorized users and Data sharing among users of the cloud. The schemes of access control can be either bitmap based or cryptographic in nature. We follow the cryptographic methods of providing access control to users to make use of cloud data. In this proposed system we have used computation based security for providing cryptographic access control of data. The Computation based cryptography makes it computationally infeasible to decrypt the data by unauthorized users by putting a price on the resource of computational power, rather than...
to give the system insufficient information to decrypt data. One such method is the RSA algorithm which is a Public Key Encryption technique which makes it infeasible to guess the Private Key by Discrete logarithm problem to prime factorization for the product of two huge primes. We also use the MVC architecture to separate the Business model and the Consumer model. The Model is operated on a list of services in the Controller to generate the View for every client.

The cloud architecture can be implemented in many ways like the Service Oriented Architecture (SOA) model and REST model. A variety of software architectural pattern to create the model like the Model View Controller (MVC) and other Rest Controller based models. Discussed some schemes to create a secure and dependable storage model. Creation of some basic authenticating schemes allows authentication. Cryptographically securing the cloud and connections via Diffie-Hellman techniques and other Security Provider services like X.509 to manage digital certificates and Key Distribution Centers (KDC) have been discussed and improved upon greatly by in.

Some methods to manage signatures and anonymous authentication of users, if necessary in the cloud are also discussed. Cipher policy based Attribute Based Encryption has been discussed by in. The use of RSA in ubiquitous computing and a Distributed Environment has been discussed.

### 2. Security and Privacy

Security and privacy are interlinked in the cloud. On one hand the user data should be kept private. On the other hand, the cloud performs a variety of checks on the data to ensure the integrity of the cloud and to check for signs of tampering. Trust upon cloud services has been discussed in Trust Cloud. The solution to such a scenario is to allow the cloud services to work only on encrypted data to prevent a breach of privacy. The cloud only needs to check the cryptographic checksums of the encrypted data to ensure its integrity and doesn’t need to know the data itself. One way to ensure this is to limit the control given to the services and define interactions among them. An external Key Distribution Center (KDC) can be setup to manage public keys only. Encrypting and Decrypting services can have a close interaction with the KDC and data transfer among users can be done via Public Key Encryption. This allows us to decentralize the model and delegate responsibility to the users. Thus, the user can rely on access control of the data.

### 3. Mathematical Background

In this section, we have the assumptions made in this paper. We also present the notations used throughout the paper in Table 1.

#### Table 1. Notations

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Meanings</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_i$</td>
<td>User of Index X</td>
</tr>
<tr>
<td>$\mathcal{U}$</td>
<td>Set of Users</td>
</tr>
<tr>
<td>$A_{ix}$</td>
<td>Attribute of $M_i$ to $U_x$</td>
</tr>
<tr>
<td>$\mathcal{A}$</td>
<td>Set of all attributes</td>
</tr>
<tr>
<td>$M_i$</td>
<td>Plaintext of index $i$</td>
</tr>
<tr>
<td>$C_{xi}$</td>
<td>Cipher encrypted with $p_{ux}$ and having plaintext $M_i$</td>
</tr>
<tr>
<td>$p_{ux}$</td>
<td>Public key of $U_x$</td>
</tr>
<tr>
<td>$p_{x}$</td>
<td>Private key of $U_x$</td>
</tr>
</tbody>
</table>

#### 3.1 Assumptions

- The cloud doesn’t want to know the plaintext of ciphers, but will perform non-invasive integrity checks without tampering with the data. This helps the cloud’s activity to remain undetected.
- All communication between the users is secured by SSL protocol.
- There are different keys for encryption and decryption.

#### 3.2 RSA Mathematical Background

RSA uses asymmetric key encryption which assigns all users with different keys for encrypting and decrypting. It is shown in Figure 1.

**Fermat’s Little Theorem:** $a^p = a \pmod{p}$, which significantly saves time and reduces space complexity in prime number finding and optimality testing by Miller Rabin’s Algorithm of huge primes probabilistically.

**Euler’s Totient function** ($\phi(n)$): The cardinality count of the reduced set of residues of $n$. In prime numbers $\phi(n) = n - 1$. In product of two primes $p$, $q$: $\phi(p \times q) = (p - 1)(q - 1)$

**Chinese Remainder Theorem:** Which allows modulo multiplication to be split into smaller fragments and used in conjunction with Fermat’s Little Theorem to encrypt data.
3.3 Public Key Encryption

The following represents the computational complexity and is easily made impossible to break by increasing bit lengths using TRNG.

3.3.1 RSA Mathematical Model

Initialization: We set the bit length as $z$. We select two large random primes from the Abelian group $p, q \in \mathbb{P}$ such that $|p|=|q|=z/2$, where $|a|$ represents the digit count of the number.

We use Miller Rabin’s algorithm to find probabilistic primes of the desired length with a Secure Random Number Generator.

We get $n = p \times q$, which belongs to $\mathbb{P} \times \mathbb{P}$. We calculate $\varphi (n) = (p-1)(q-1)$, where $\varphi (n)$ is Euler’s totient function. Then we find a number $e$ such that $\gcd(\varphi (n), e) = 1, 1 < e < n$.

Also calculate $d = \text{mod} \_\text{Inverse} (m, e)$ by Extended Euclid’s theorem, which is the solution to the equation $e \cdot d = 1 \mod \varphi (n)$, or $d = e^{-1} \mod n$.

Publish $Pu = (n, e)$ and store $Pr = (n, d)$.

3.3.2 Encryption via RSA:

To encrypt a message $m$, convert it to a byte array and apply

$$(m^e) \mod n = c$$

where $c = \text{cipher}$

3.3.3 Decryption via RSA:

To decrypt

$$(c^d) \mod n = m$$

where $m = \text{message}$

The implementation of RSA is shown in Figure 2.

4. SCACO System Model

The Secure Cloud-based Access Control Optimization and its implementation is shown in Figure 3 and Figure 4, respectively.
4.1 Cloud User Initialization
Generate $Pu_A$, $Pr_A$, keys for users $U_x \in \mathcal{U}$, on registration by RSA algorithm.

User $A$ encrypts message via RSA to generate cipher,

$$C_{BI} = \text{Encrypt-RSA} (Pu_B, M_I).$$

SCACO cannot decrypt the message.

Receive Ciphers from $A$.

4.2 Server Attribute Generation
Assume message $M_I$ is to be sent by $U_A$ to $U_B$. Generate attribute,

$$A_{BI} = \text{substring} (\text{SHA-2} (M_I, Pu_B))$$

and send to $B$.

The attribute should depend on the message and be recipient specific. The attribute is made small to allow easy entry by humans. We used 5 digit hexadecimal codes as attributes. SCACO Attribute based System Model along with its common cloud are shown in Figure 5 and Figure 6.

4.3 Server Distribution of Data
Verify $(A_{BI}) \in \mathcal{A}$ sent by $B$.

Send $C_{BI}$ to $B$ via SSL or SSH depending on authentication requirements.

$U_B$ can decrypt,

$$M_{BI} = \text{Decrypt-RSA} (Pr_B, C_{BI}).$$
4.4 Advantages of SCACO

SCACO cannot decrypt the ciphers it stores.
- The user can carry list of attributes whose size is much smaller and request all ciphers multiple times.
- RSA prevents modification of ciphers stored on SCACO.
- SCACO can perform integrity checks on data.

5. Security

RNG: Random number Generator used should abide by FIPS 140-2, Security Requirements for Cryptographic Modules and RFC 1750: Randomness Recommendations for Security.

Discrete logarithm: The inverse problem to exponentiation is the discrete logarithm problem, which allows us to find primitive roots, solving the prime factorization problem of two huge primes. The finding of discrete logarithm is an NP-hard problem. Some solutions find discrete logarithm in \( O(n \log n + n \sqrt{p}) \) time, which is still unfeasible to decrypt the public key of a user.

Possible approaches to attacking RSA are:
- Brute force key search (infeasible given size of numbers)
Mathematical attacks (based on difficulty of computing $\phi(n)$, by factoring modulus $n$)
Chosen cipher text attacks (given properties of RSA)
Keeping these in mind we should increase the bit length with time to keep the complexity high.

5.1 Computation Complexity

$K$ is constant $\text{bit\_length}$.

Public key operations take $O(k^2)$ steps.

Private key operations take $O(k^3)$ steps.

Key generation takes $O(k^4)$ steps, where $k$ is the bit\_length($n$).

Note: $\log d = \text{Approx.} (\log n)$ and $O(\log e) = 1$.
Therefore all operations run in Approx. $O(1)$ for small $k$.
Approx. 35ms for each operation, $k=1024$.

6. Conclusion

We have implemented a secure cloud model based on Public Key Encryption. The cloud is decentralized into User Clouds with RSA keys. Attributes are generated to provide cipher permissions to the correct user. Integrity checks are performed to prevent tampering by Message Digests. One limitation of this model is that Multiparty Signatures are not generated by the same mathematical equation, rather divided into individual attributes. We would like to provide such a future enhancement in SCACO.

7. References