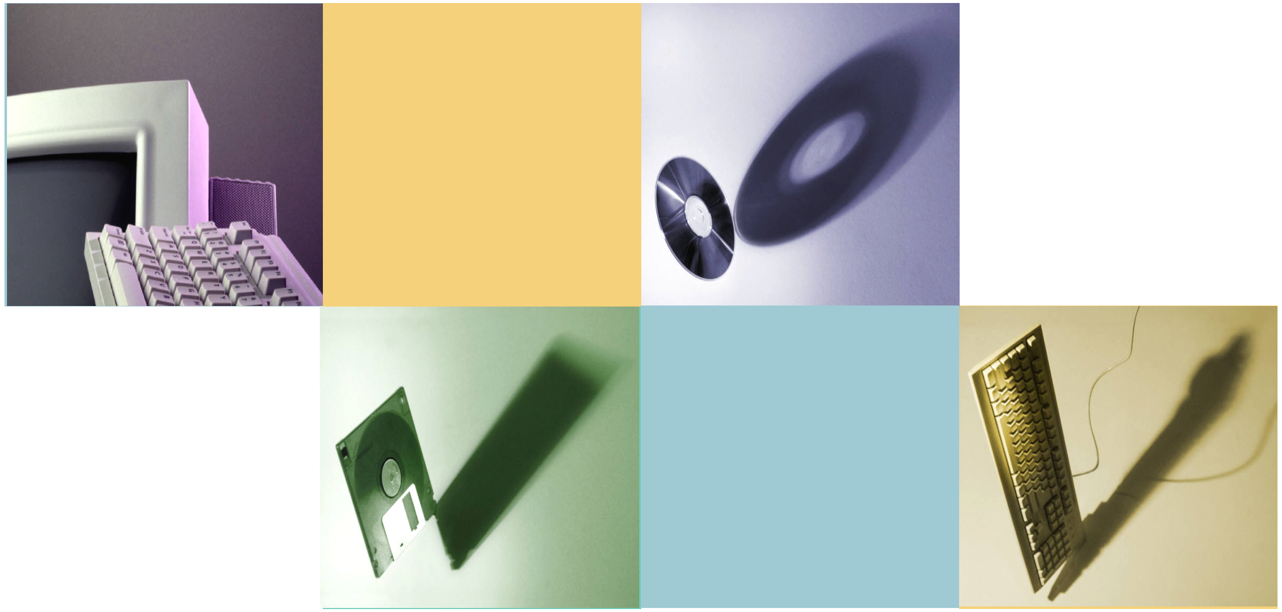
The 4th International Workshop on Security in Cloud Computing (CloudSec 2012)

Clock-Based Proxy Re-encryption Scheme in Unreliable Clouds

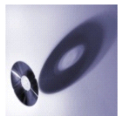
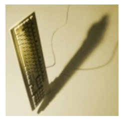


Qin Liu[1][2], Guojun Wang[1], and Jie Wu[2] ,

[1] Central South University, China
[2] Temple University, USA

Outline

1. Introduction
2. Background
3. Scheme description
4. Conclusion

Introduction

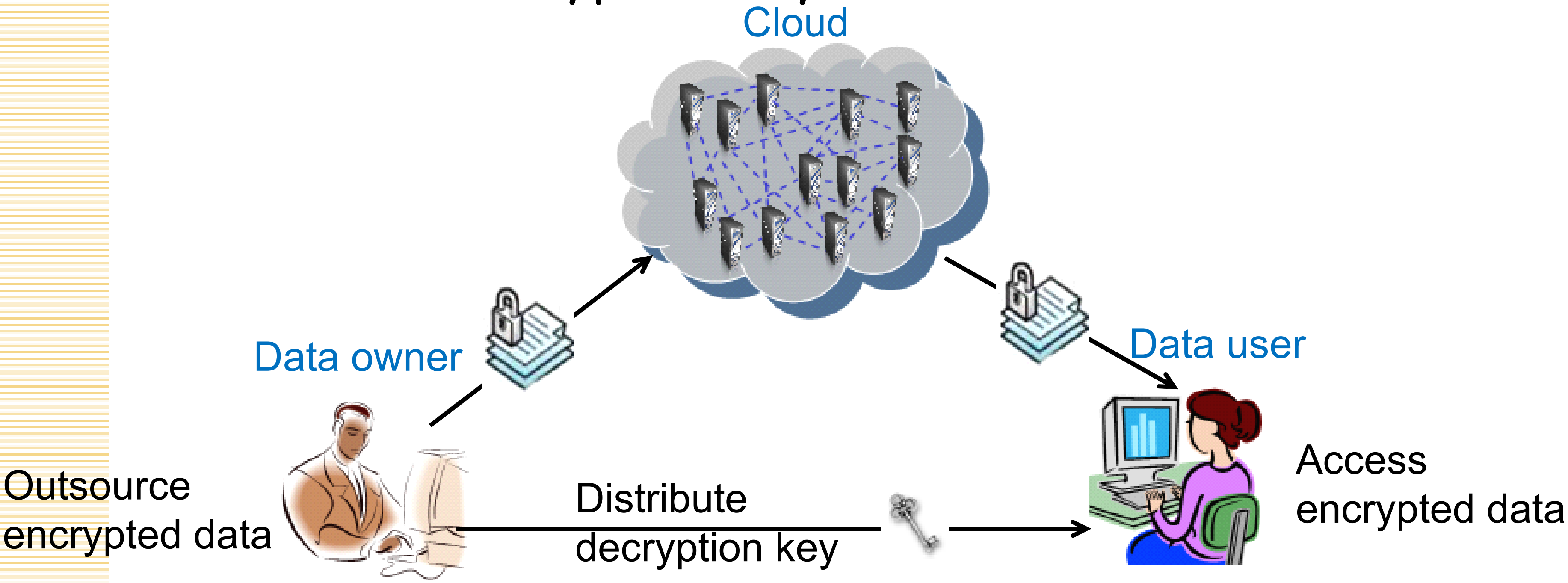
- Cloud computing
- Cloud security
- Challenge——User revocation
- New challenge in unreliable cloud

Cloud computing

○ Cloud computing has emerged as a new type of commercial paradigm due to its overwhelming advantages, such as flexibility, scalability, and cost efficiency.

Cloud security

- One technique to protect the data from a potentially untrusted cloud service provider (CSP) is for the data owner to encrypt data and distribute decryption key to authorized data users.



Challenge——User revocation

○ The key problem of storing encrypted data in the cloud lies in *revoking access rights from users*.

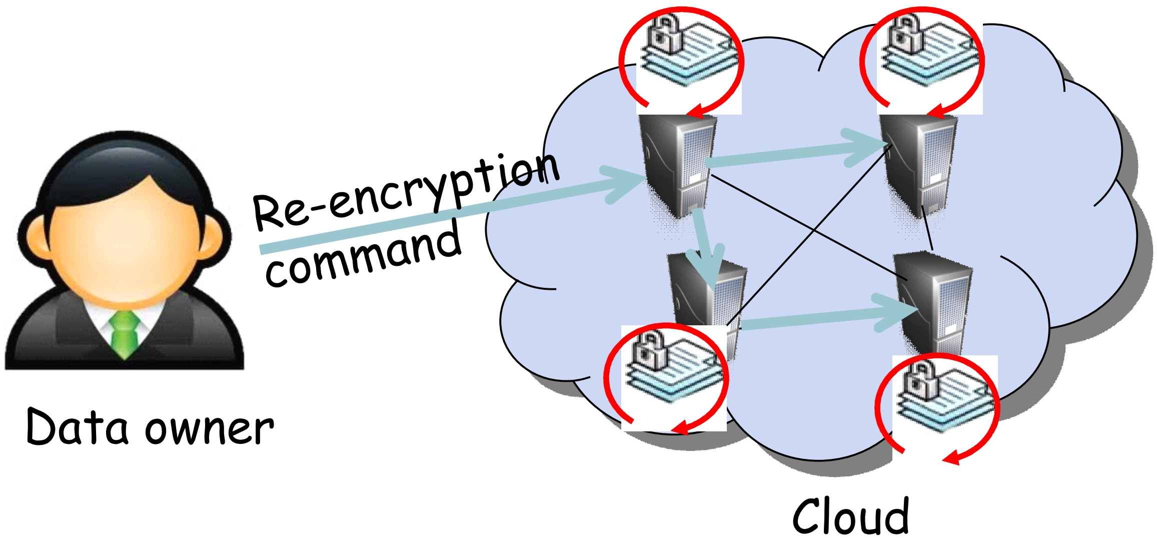
○ *A user whose permission* is revoked will still retain the keys issued earlier, and thus can still decrypt data in the cloud.

○ Data owner should:
- ○(1) Re-encrypt data
- ○(2) Re-key to remaining users

○Frequent revocation
→performance bottleneck

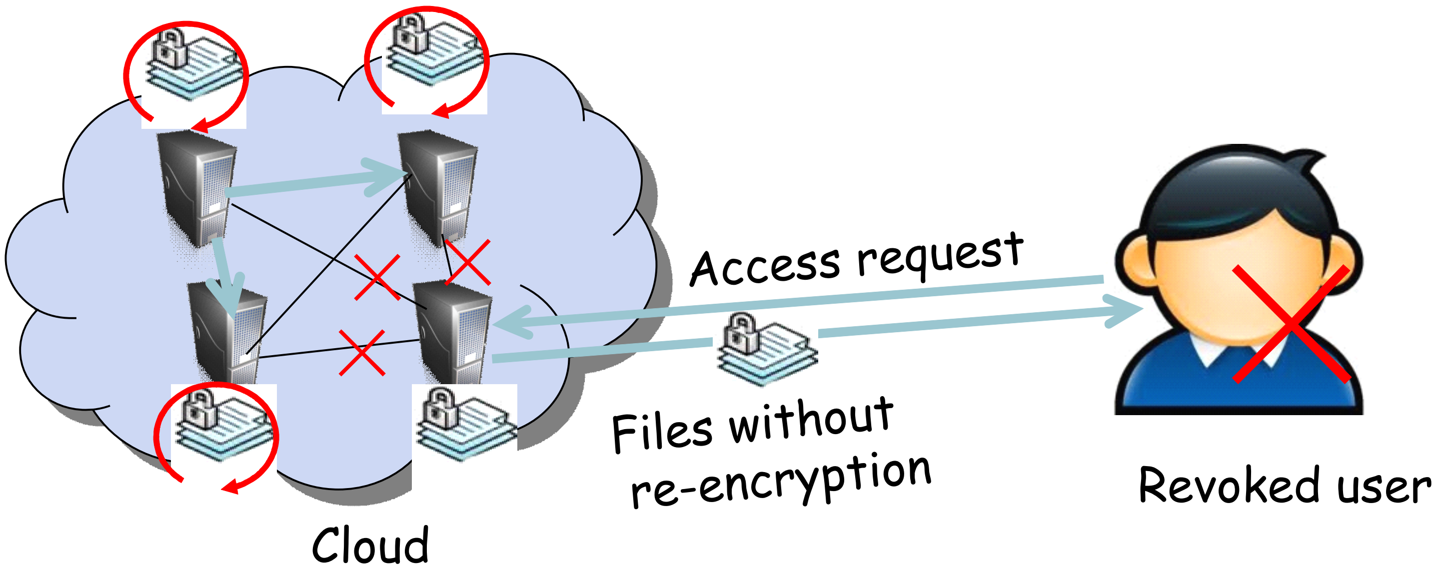
New challenge in unreliable clouds

- Data is replicated over multiple servers for high availability. Cloud servers execute re-encryption while receiving commands.



New challenge in unreliable clouds

○ While experiencing network outages, commands cannot propagate to all servers in a timely fashion, thus creating security risks.

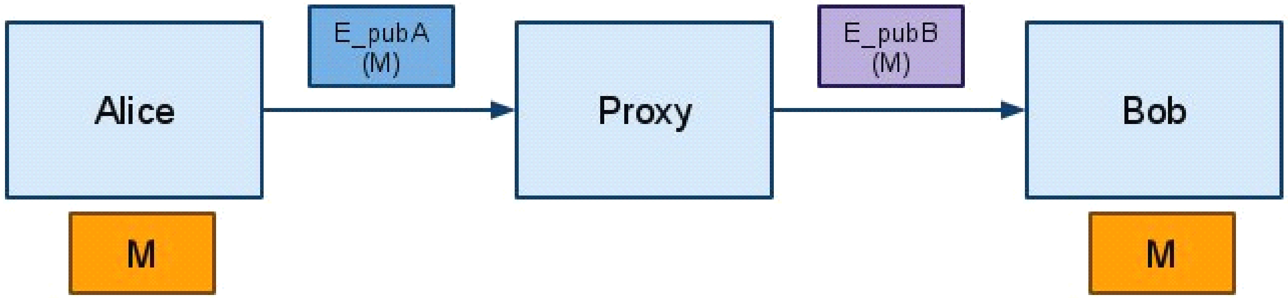


Background

- Proxy re-encryption (PRE)
- Attribute-based encryption (ABE)
- System model
- Adversary model

Proxy re-encryption (PRE)

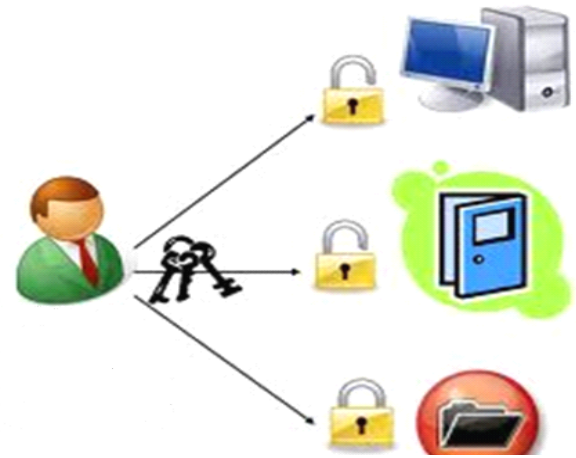
- To reduce the workload at the data owner, *proxy re-encryption* (PRE) technique is applied to delegate the cloud for re-encryption.



PRE allows the cloud to convert a ciphertext encrypted under Alice's public key into the ciphertext that can be decrypted by Bob's private key without seeing the underlying plaintext.

Attribute-based encryption (ABE)

ABE allows to encrypt data specifying an access control policy over *attributes*, so that only users with a set of attributes satisfying this policy can decrypt the corresponding data



For example, a file encrypted using the access structure $(a_1 \wedge a_2) \vee a_3$ means that either a user with attributes a_1 and a_2, or a user with attribute a_3, can decrypt the file.

System model

- The data owner outsources a set of files $F_1, \ldots, F_n$ *to the cloud.*

- Each file *is encrypted with two parameters, access time* and *access structure*.

- Each user is associated with a set of attributes and an eligible time, where the eligible time means how long the user can access the data.

System model

○ The data owner and the cloud share a root secret key s in advance, so that the cloud can use s to calculate the PRE keys based on its internal clock, and re-encrypts the ciphertext with these PRE keys.

○A file can be decrypted by only the users whose attributes satisfy the access structure, and whose eligible time satisfies the access time.

Adversary model

- Cloud service provider (CSP)
  - Honest-but-curious: Always correctly execute a given protocol, but may try to gain some additional information about data.

- Malicious data users
  - Try to learn the file content that he is not authorized to access.

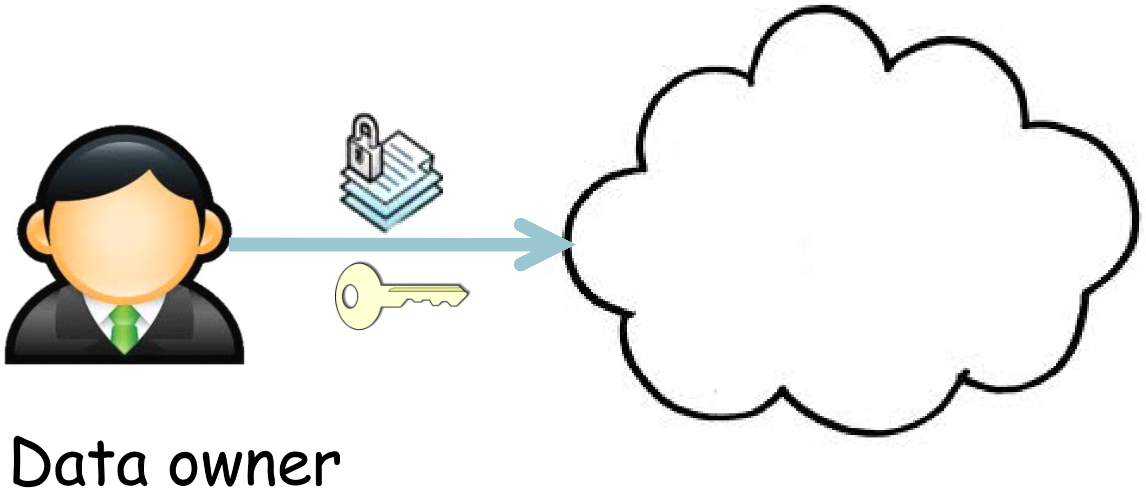
- CSP and data users will not collude

Scheme description

- Intuition
- Scheme definition
- Scheme construction

Intuition

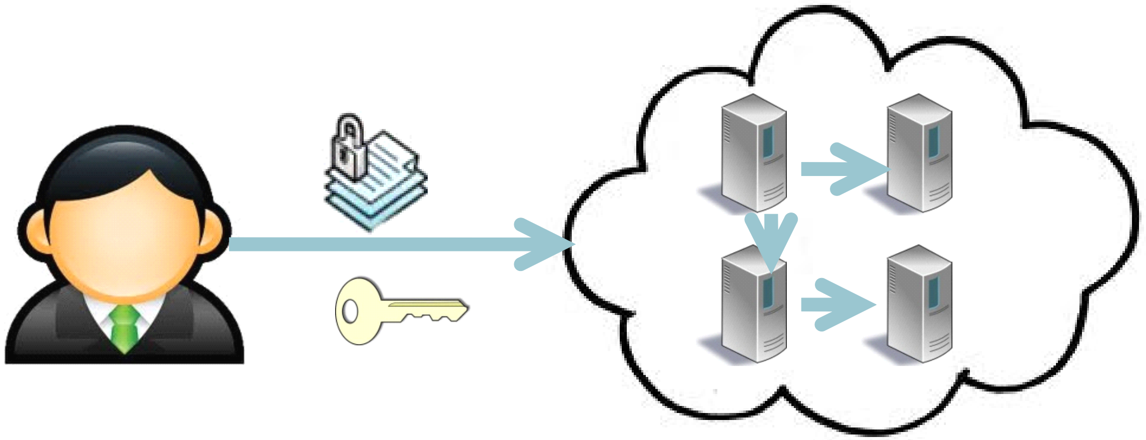
- The data owner sends the encrypted file and a root secret key to the CSP



- The shared secret key
  - Cannot decrypt the file
  - is used to re-encrypt file

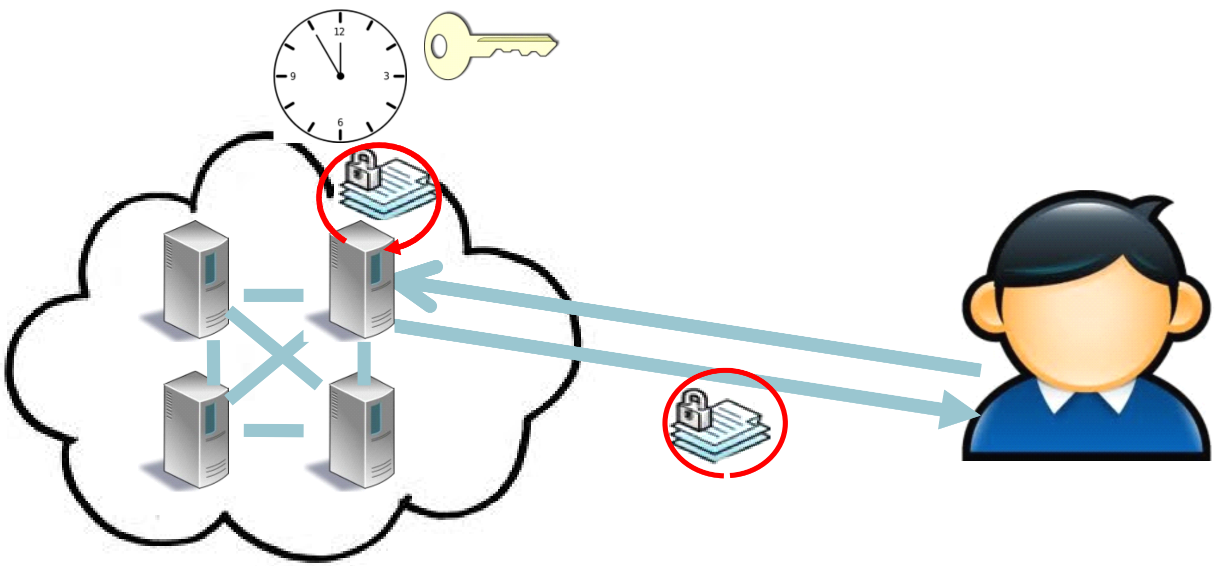
Intuition

- The CSP will replicate the file as well as the root secret key to many cloud servers.



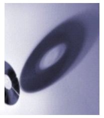
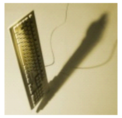
Intuition

- When receiving a request from a user, the cloud server automatically re-encrypt the file using the root secret key based on its own clock



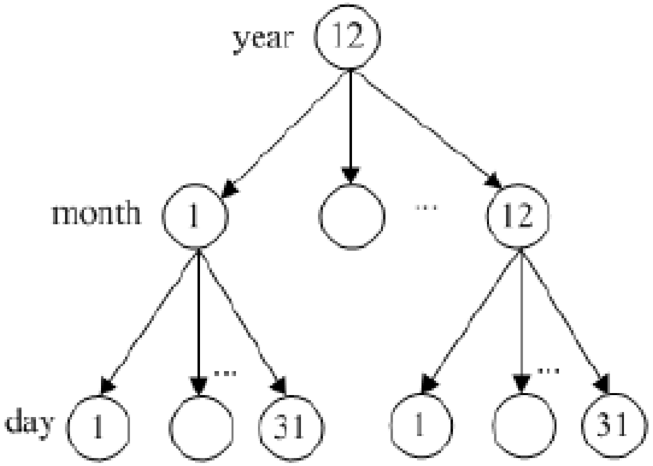
Scheme Definitions

1. $Setup(K, \mathbb{UA}) \rightarrow (PK, MK, s)$: The data owner takes a sufficiently large security parameter K as input to generate the system public key PK, the system master key MK, and the root secret key s. The system public key will be published, the system master key will be kept secret, and the root secret key will be sent to the CSP.
2. $GenKey(PK, MK, s, PK_u, a, T_u) \rightarrow (SK_u, SK_{u,a}^{T_u})$: Suppose that user u with public key PK_u is eligible for attribute a and his access right is effective in time T_u. The data owner uses the system public key PK, the system master key MK, the root secret key s, user public key PK_u, attribute a, and effective time period T_u to generates user identity secret key (UIK) SK_u and time-based user attribute secret key (UAK) $SK_{u,a}^{T_u}$ for u.
3. $Encrypt(PK, \mathbb{A}, F) \rightarrow (C_{\mathbb{A}})$: The data owner takes a DNF access structure $\mathbb{A}$, a data F, and system public key PK, e.g., initial public keys of all attributes in the access structure $\{PK_a\}_{a \in \mathbb{A}}$ as inputs to output a ciphertext $C_{\mathbb{A}}$.
4. $ReEncrypt(C_{\mathbb{A}}, PK, s, t) \rightarrow (C_{\mathbb{A}}^t)$: Given a ciphertext $C_{\mathbb{A}}$ with structure $\mathbb{A}$, the CSP uses the system public key PK, the root secret key s, and the access time t to re-encrypt the original ciphertext $C_{\mathbb{A}}$ to $C_{\mathbb{A}}^t$.
5. $Decrypt(PK, C_{\mathbb{A}}^t, SK_u, \{SK_{u,a}^{T_u}\}_{a \subseteq \mathbb{A}, T_u \subseteq t}) \rightarrow (F)$: User u, whose attributes satisfy the access structure $\mathbb{A}$, and whose effective time period T_u satisfy the access time t, can use SK_u and $\{SK_{u,a}^{T_u}\}_{a \subseteq \mathbb{A}}$ to recover F from $C_{\mathbb{A}}^t$.

Scheme Construction

- Time is divided into a time tree.



Sample time tree

We use (y, m, d), (y,m), and (y) to denote a particular day, month, and year, respectively.
For example, (2011, 4, 5) denotes April 5, 2011.

For each attribute a, the data owner calculates the PRE keys in a hierarchical way: $s_a = H_s(PK_a)$, $s_a^y = H_{s_a}(y)$, $s_a^{y,m} = H_{s_a^y}(m)$, and $s_a^{y,m,d} = H_{s_a^{y,m}}(d)$, where PK_a is attribute a's public key; y, m, and d denote a specific year, month, and day, respectively; and $H_s, H_{s_a}, H_{s_a^y}$, and $H_{s_a^{y,m}}$, are hash functions with indexes s, s_a, s_a^y, and $s_a^{y,m}$, respectively.

Scheme Construction

1. $Setup(K) \rightarrow (PK, MK, s)$: The data owner takes security parameter K as input to generate the system public key PK, the system master key MK, and the secret shared key s. Specifically, he first defines the universe attributes $\mathbb{UA}$. Then, for each attribute a in $\mathbb{UA}$, he generates a public/private key pair (sk_a, PK_a), where sk_a is randomly chosen from $\mathbb{Z}_q$ and $PK_a = sk_a P_0$. Next, he computes $Q_0 = mk_0 P_0$ and $SK_1 = mk_0 P_1$, where mk_0 is randomly chosen from $\mathbb{Z}_q$ and P_1 is randomly chosen from $\mathbb{G}_1$. Finally, he randomly chooses mk_1 and s from $\mathbb{Z}_q$, publishes $\{PK_a\}_{a \in \mathbb{UA}}$ and $(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P_0, P_1, Q_0)$ as system public key PK, keeps $\{sk_a\}_{a \in \mathbb{UA}}$, mk_0, mk_1, and SK_1 as system master key MK, and sends s to the cloud as a shared secret key.

Scheme Construction

2. $GenKey(PK, MK, s, PK_u, A_u, T_u) \rightarrow (SK_u, \{SK_{u,a}^{T_u}\}_{a \in A_u})$: Suppose user $\mathcal{U}$ with public key PK_u possesses an attribute set A_u with eligible time T_u. Then, the data owner first generates a user identity secret key SK_u, and then for each attribute $a \in A_u$, he generates a user attribute secret key $SK_{u,a}^{T_u}$ for $\mathcal{U}$. Specifically, the data owner calculates user master key with $mk_u = H_{mk_1}(PK_u)$, and sets $SK_u = mk_1 mk_u P_0$, where $H_{mk_1} : \mathbb{G}_1 \rightarrow \mathbb{Z}_q$. For each attribute $a \in A_u$, he sets $SK_{u,a}^{T_u} = SK_1 + mk_1 mk_u (PK_a + s_a^{T_u} P_0)$, where $s_a^{T_u}$ is the PRE key on attribute a in time T_u. Note that if T_u is a particular year (y), then $s_a^y = H_{s_a}(y)$; if T_u is a particular month (y, m), then $s_a^{y,m} = H_{s_a^y}(m)$; if T_u is a particular day (y, m, d), then $s_a^{y,m,d} = H_{s_a^{y,m}}(d)$, where $s_a = H_s(PK_a)$, $H_s : \mathbb{G}_1 \rightarrow \mathbb{Z}_q$, and $H_{s_a}, H_{s_a^y}, H_{s_a^{y,m}} : \{0,1\}^* \rightarrow \mathbb{Z}_q$.

Scheme Construction

3. $Encrypt(PK, \mathbb{A}, D) \rightarrow (C_{\mathbb{A}})$: Given an access structure $\mathbb{A} = \bigvee_{i=1}^{N} (CC_i) = \bigvee_{i=1}^{N} (\bigwedge_{j=1}^{n_i} a_{ij})$ that is in the disjunctive normal form (DNF), the data owner encrypts data D as follows: He first picks a random element $r \in \mathbb{Z}_q$, and then sets $n_{\mathbb{A}}$ to be the lowest common multiple (LCM) of $n_1, \ldots, n_N$. Finally, he calculates Eq. (1) to produce the ciphertext:

$$\begin{aligned} &U_0 = rP_0, V = D \cdot \hat{e}(Q_0, rn_{\mathbb{A}}P_1) \\ &\{U_i = r \sum_{a \in CC_i} PK_a\}_{1 \leq i \leq N} \end{aligned} \qquad (1)$$

The ciphertext is set to $C_{\mathbb{A}} = (\mathbb{A}, U_0, \{U_i\}_{1 \leq i \leq N}, V)$.

Scheme Construction

4. $ReEncrypt(C_{\mathbb{A}}, s, t) \rightarrow C^t_{\mathbb{A}}$: On receiving the user's request for data D, the cloud first determines current time $t = (y, m, d)$. Then, it randomly chooses $r' \in \mathbb{Z}_q$, and re-encrypts data D with Eq. (2):

$$
\begin{gathered}
U_0^t = U_0 + r' P_0, V^t = V \cdot \hat{e}(Q_0, r' n_{\mathbb{A}} P_1) \\
U^t_{(y)i} = \sum_{a \in CC_i} (U_i + r' PK_a + s_a^y U_0^t) \\
U^t_{(y,m)i} = \sum_{a \in CC_i} (U_i + r' PK_a + s_a^{y,m} U_0^t) \\
U^t_{(y,m,d)i} = \sum_{a \in CC_i} (U_i + r' PK_a + s_a^{y,m,d} U_0^t)
\end{gathered} \tag{2}
$$

The ciphertext is $C^t_{\mathbb{A}} = (V^t, \mathbb{A}, t, U_0^t, \{U^t_{(y)i}, U^t_{(y,m)i}, U^t_{(y,m,d)i}\}_{1 \leq i \leq N})$.

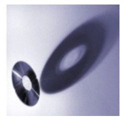
Scheme Construction

5. $Decrypt(PK, C^t_{\mathbb{A}}, SK_u, \{SK^{T_u}_{u,a}\}_{a \subseteq \mathbb{A}, T_u \subseteq t}) \rightarrow D$: Given ciphertext $C^t_{\mathbb{A}}$, user $\mathcal{U}$, whose attributes satisfy the access structure $\mathbb{A}$, e.g., possessing all attributes in the *i-th* conjunctive clause CC_i, and the eligible time T_u satisfies t, computes Eq. (3) to recover D:

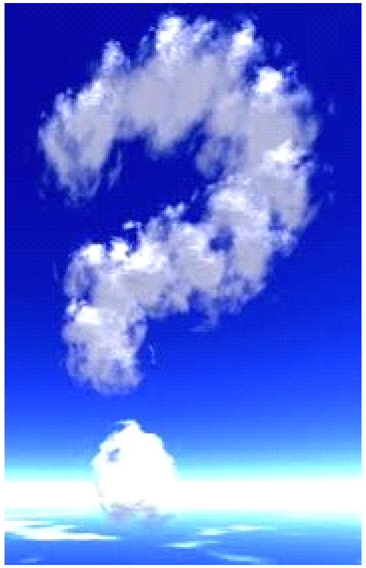
$$V^t / \left(\frac{\hat{e}(U^t_0, \frac{n_{\mathbb{A}}}{n_i} \sum_{a \in CC_i} SK^{T_u}_{u,a})}{\hat{e}(SK_u, \frac{n_{\mathbb{A}}}{n_i} U^t_{(T_u)i})} \right) \quad (3)$$

Conclusion

1. An automatic, time-based, re-encryption scheme is proposed for unreliable cloud environments
2. An attribute-based encryption (ABE) scheme is extended by incorporating timestamps to perform proxy re-encryption
3. We incorporate time concept to the encryption scheme, so that the user with a small number of secret keys can rapidly recover data


Thank you!



I can pass on your
questions to Ms. Qin Liu