Towards Temporal Access Control in Cloud Computing

Yan Zhu*, Hongxin Hu†, Gail-Joon Ahn†, Dijiang Huang†, and Shanbiao Wang*
* Peking University, Beijing, 100080, China
† Arizona State University, Tempe, Arizona, 85281, USA
{yan.zhu,wangshanbiao}@pku.edu.cn; {hxhu,gahn,dijiang.huang}@asu.edu

***Abstract*—Access control is one of the most important security mechanisms in cloud computing. Attribute-based access control provides a flexible approach that allows data owners to integrate data access policies within the encrypted data. However, little work has been done to explore temporal attributes in specifying and enforcing the data owner's policy and the data user's privileges in cloud-based environments. In this paper, we present an efficient temporal access control encryption scheme for cloud services with the help of cryptographic integer comparisons and a proxy-based re-encryption mechanism on the current time. We also provide a dual comparative expression of integer ranges to extend the power of attribute expression for implementing various temporal constraints. We prove the security strength of the proposed scheme and our experimental results not only validate the effectiveness of our scheme, but also show that the proposed integer comparison scheme performs significantly better than previous bitwise comparison scheme.**

***Index Terms*—Cryptography, Temporal Access Control, Re-Encryption, Integer Comparison, Cloud Computing**

I. Introduction

Cloud computing provides an extensible and powerful environment for growing amounts of services and data by means of on-demand self-service. It also relieves the client's burden from management and maintenance by providing a comparably low-cost, scalable, location-independent platform. However, cloud computing is also facing many challenges for data security as the users outsource their sensitive data to clouds, which are generally beyond the same trusted domain as data owners.

To address this problem, access control is considered as one of critical security mechanisms for data protection in cloud applications. Unfortunately, traditional data access control schemes usually assume that data is stored on trusted data servers for all users. This assumption however no longer holds in cloud computing since the data owner and cloud servers are very likely to be in two different domains. Hence, attribute-based access control [1], [2] has been introduced into cloud computing to encrypt outsourced sensitive data in terms of access policy on attributes describing the outsourced data, and only authorized users can decrypt and access the data. Since the access control policy of every object is embedded within it, the enforcement of policy becomes an inseparable characteristic of the data itself. This is in direct contrast to most currently available access control systems, which rely directly upon a trusted host to mediate access and maintain policies.

Even though there have been some previous work to construct fine-grained access control systems in clouds [3], [4], existing work lacks a systematic mechanism to support a complete temporal control. Temporal dimension has generated a great amount of interest in security community as an important property of access control for security system management in recent years [5], [6]. However, existing attribute-based solutions are difficult to provide full features of temporal data access control due to following reasons:

- The system models of existing systems cannot support dual comparative expressions (DTE), in which two range-based comparative constraints must be embedded into the outsourced files as well as the user's private key.
- The existing systems don't support current time, which is essentially an important factor for enforcing temporal access control.
- Bethencourt *et al.* [1] has provided a bitwise-comparison method (called BSW's scheme) to realize a pretty simple control, e.g. $a < 11$, but this method does not support range expressions in user's private key because both "*1*" and "*0*" may appear in the same bit position.

In this paper, we address the afore-mentioned problems by constructing a temporal access control solution along with a proxy-based re-encryption mechanism [7] for cloud computing. The proposed scheme is originated from the needs of practical cloud applications, in which each outsourced resource can be associated with an access policy on a set of temporal attributes, e.g., period-of-validity, opening hours, or hours of service. Each user can also be assigned a license with several privileges based on the comparative attributes. To enforce the valid matches between access policies and user's privileges, we introduce a proxy-based re-encryption mechanism [7] with respect to the current time. This design brings about several efficient benefits, such as flexibility, supervisory, and privacy protection, compared with prior work.

Our solution also addresses another practical issue to implement cryptographic integer comparisons and re-encryption mechanism on the current time. We provide a cryptographic expression of integer ranges to extend

Y. Zhu works in Beijing Key Laboratory of Internet Security Technology and Institute of Computer Science and Technology, Peking University, Beijing, 100080, China. This work of Y. Zhu and S. Wang was supported by the National Natural Science Foundation of China (Project No. 61170264 and 10990011).

This work of G.-J. Ahn and H. Hu was partially supported by the grants from US National Science Foundation (NSF-IIS-0900970 and NSF-CNS-0831360) and Department of Energy (DE-SC0004308). D. Huang's research is sponsored by Office of Naval Research Young Investigator Program (ONR-YIP) and NSF-CNS-1029546.

978-1-4673-0775-8/12/$31.00 ©2012 IEEE

the power of attribute expression, and propose a temporal access control encryption (TACE) scheme to implement various temporal constraints. This scheme provides a constant size of ciphertext, private-key, and depth of policy-tree, as well as a nearly linear-time complexity. Other security features, such as forward and backward derivation functions, are provided in our scheme as well. In addition, we prove the security of these two functions under the RSA and Co-CDH assumption [8]. To demonstrate the feasibility of our proposed approach, we implement a prototype of TACE system. Our experimental results not only validate the effectiveness of our scheme and algorithms, but also show our scheme has better performance for integer comparison than existing bitwise comparison scheme.

This paper is organized as follows. Section II discusses our research goals and models. Section III shows our framework and security requirements. Section IV provides main techniques pertaining to our construction. In Section V, we analyze our scheme in terms of its security and performance, respectively. Finally, we discuss the related work in Section VI and conclude this paper in VII.

II. Problem Statement

A. Design Goals

Our main design goal is to help the data owner achieve temporal data access control on files stored in cloud servers. Although this kind of access control is based on fine-grained access control introduced for outsourced data services [3], we intent to ensure that all kind of temporal access policy can be securely and efficiently implemented for outsourced data services. Specially, we also want to solve the following problem: *Given an access constraint $t_i \leq A_t \leq t_j$ in policy $\mathcal{P}$ embedded in the ciphertext $\mathcal{C}$ and a privilege $t_a \leq A_t \leq t_b$ in the user's private key SK, how to guarantee that the ciphertext can be decrypted only at a valid current time t_c?* Here, the valid current time t_c means that two conditions, $t_c \in [t_i, t_j]$ and $t_c \in [t_a, t_b]$, must be satisfied at the same time.

B. System Model

Considering a cloud-based data storage service involving three different entities, as illustrated in Fig. 1: data owner, cloud server, and many data users (e.g., computers, mobile devices, or general equipments). In addition, in order to implement temporal access control, we require a clock server designed to always provide exactly the same current time by communicating with each other.

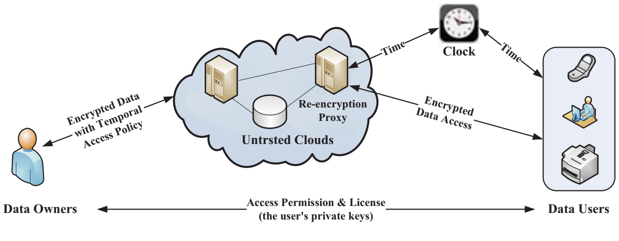

Clock
Time
Time
Encrypted Data with Temporal Access Policy
Re-encryption Proxy
Untrsted Clouds
Encrypted Data Access
Data Owners
Access Permission & License (the user's private keys)
Data Users

Fig. 1. Temporal constraints on privilege-licence assignment.

To ensure the data access compliant with the assigned policy, fine-grained access control has been introduced into the outsourced storage service. We extend this kind of access control mechanisms to support temporal access control encryption (TACE) described as follows:

- First, the data owner makes use of a temporal access policy $\mathcal{P}$ to encrypt data before store it to clouds.
- Second, once receiving an access request from a user, the cloud service checks whether corresponding temporal constraints can be satisfied in $\mathcal{P}$ with respect to the current time t_c, then employs a re-encryption method to convert the encrypted data into another ciphertext C_{t_c} that embed current time t_c and sent it the user.
- Finally, the authorized user can use her/his private key SK with access privilege $\mathcal{L}$ to decrypt C_{t_c}. In this model, we assume the cloud service is a semi-trusted service that can use the correct time to re-encrypt data.

C. Benefits of TACE

Flexibility: TACE-based cryptosystem can provide more flexible access control based on temporal constraints as follows: a) Date control on Year, Month, and Day, e.g., $((2010 \leq Year \leq 2011)$ AND $(4 \leq Month \leq 7))$; and b) Periodic control on Week and Hour, e.g., $((3 \leq Week \leq 5)$ AND $(8:00PM \leq Hour \leq 10:00PM))$. More importantly, this cryptosystem also supports all kind of level controls and integer comparisons, e.g., $((3 \leq \text{Security Clearance} \leq 5)$ OR $(2,000 \leq \text{Salary} \leq 5,000))$.

Supervisory: Traditional cryptosystems, that only contains both encryption and decryption processes, has not an efficient method to monitor the usage of encrypted data. TACE-based cryptosystem introduces a proxy-based re-encryption mechanism that can apply the current time to determine whether the user's download request is reasonable, and rely on the re-encryption technologies to produce a new version of data under the current time. Such a proxy service can also integrate with other rich information to determine the legitimacy of user behaviors.

Privacy Protection: In our system model, the access policies are enforced entirely dependent upon temporal attribute matches between ciphertexts and private keys in the client side. In the re-encryption process, cloud servers do not require any user information which is used to enforce access policies. Hence, this mechanism ensures that user privacy, including user identity and access privilege in the user's private key, will not be disclosed to cloud servers.

III. Framework and Security Requirements

A. Notations

For sake of clarity, we introduce following notations:

- $\mathcal{A}$: the set of attributes $\mathcal{A} = \{A_1, \cdots, A_m\}$;
- $A_k(t_i, t_j)$: the range constraint of attribute A_k on $[t_i, t_j]$, i.e., $t_i \leq A_k \leq t_j$;
- $\mathcal{P}$: the access control policy expressed as a Boolean function on AND/OR logical operations, generated by the grammar: $\mathcal{P} ::= A_k(t_i, t_j) | \mathcal{P}$ AND $\mathcal{P} | \mathcal{P}$ OR $\mathcal{P}$;

- $\mathcal{L}$: the access privilege assigned into the user's licence, generated by $\mathcal{L} ::= \{A_k(t_a, t_b)\}_{A_k \in \mathcal{A}}$.

The definitions of $\mathcal{P}$ and $\mathcal{C}$ can meet the basic requirements of dual temporal expressions. Given a time assignment t_c for A_k, the constraint or privilege $A_k(t_i, t_j)$ outputs $true$ if $t_i \leq t_c \leq t_j$, otherwise outputs $false$. We call it a valid time assignment if and only if both $A_k(t_i, t_j) \in \mathcal{P}$ and $A_k(t_a, t_b) \in \mathcal{L}$ output true.

B. TACE Framework

With focusing on temporal access control and re-encryption mechanism in cloud computing, the TACE scheme consists of five algorithms:

1) Setup($1^\kappa, \mathcal{A}$): Takes a security parameter κ and a list of attributes $\mathcal{A}$ as input, outputs the master key MK and the public-key $PK_\mathcal{A}$;
2) GenKey($MK, u_k, \mathcal{L}$): Takes the user's ID number u_k as input, the access privilege $\mathcal{L}$ and MK, outputs the user's private key $SK_\mathcal{L}$;
3) Encrypt($PK_\mathcal{A}, \mathcal{P}$): Takes a temporal access policy $\mathcal{P}$ and $PK_\mathcal{A}$ as input, outputs the ciphertext header $\mathcal{H}_\mathcal{P}$ and a random session key ek;
4) ReEncrypt($PK_\mathcal{A}, \mathcal{H}_\mathcal{P}, t_c$): Takes a current time t_c and a ciphertext header $\mathcal{H}_\mathcal{P}$ and $PK_\mathcal{A}$ as input, outputs a new ciphertext header H_{t_c};
5) Decrypt($SK_\mathcal{L}, \mathcal{H}_{t_c}$): Takes a user's private key $SK_\mathcal{L}$, and a ciphertext header $\mathcal{H}_{t_c}$ on the current time t_c as input, outputs a session key ek;

With the help of this framework, the workflow of TACE-based cryptosystem is described in Fig.2. For sake of clarity, the operations on the data are not shown in the framework since data owner could easily employ traditional symmetric key cryptography to encrypt and then outsource data with the help of a random session key.

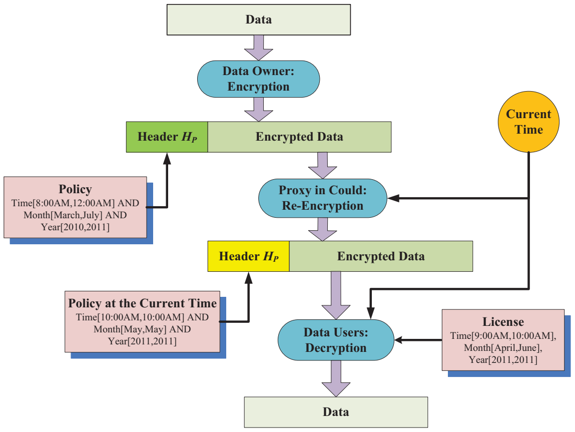

Data
Data Owner: Encryption
Header H_P
Encrypted Data
Current Time
Policy
Time[8:00AM,12:00AM] AND Month[March,July] AND Year[2010,2011]
Proxy in Could: Re-Encryption
Header H_P
Encrypted Data
Policy at the Current Time
Time[10:00AM,10:00AM] AND Month[May,May] AND Year[2011,2011]
Data Users: Decryption
License
Time[9:00AM,10:00AM], Month[April,June], Year[2011,2011]
Data

Fig. 2. Workflow of TACE-based Cryptosystem.

This framework is based on BSW's scheme [1], in which both AND/OR operations and basic fine-grained access control are not within the scope of this paper.

C. Security Models

First, given a scheme based on our TACE framework, we must guarantee that this scheme can follow the principle in secure temporal control: Let $A_k \in \mathcal{A}$ be a range-based temporal attribute and $(\mathcal{P}, \mathcal{L})$ be a constraint-privilege pair with A_k, where $A_k[t_i, t_j] \in \mathcal{P}$ and $A_k[t_a, t_b] \in \mathcal{L}$. Given a current time t_c, secure temporal control requires that the access is granted if and only if $t_c \in [t_i, t_j]$ and $t_c \in [t_a, t_b]$. This means that the TACE scheme can must also obey this rule as follows: Given the above-mentioned $(\mathcal{P}, \mathcal{L})$, we can compute $(MK, PK_\mathcal{A}) \leftarrow Setup(1^\kappa, \mathcal{A})$, $SK_\mathcal{L} \leftarrow GenKey(MK, u_k, \mathcal{L})$, and $(\mathcal{H}_\mathcal{P}, ek) \leftarrow Encrypt(PK, \mathcal{P})$. Such that, we hold

$$\Pr\left[\begin{array}{c} \mathcal{H}_c \leftarrow ReEncrypt(PK_\mathcal{A}, \mathcal{H}_\mathcal{P}, t_c); \\ Decrypt(SK_\mathcal{L}, \mathcal{H}_{t_c}) = ek \end{array}\right] = 1,$$

if and only if the access is granted over $(\mathcal{P}, \mathcal{L})$ and t_c according to fine-grained access control model. Besides these, we are more concerned with the security risk from cloud servers or data users, as follows:

- Cloud servers: Similarly to [9], [3], we just consider "Honest but Curious" cloud servers, that is, cloud servers will follow our proposed protocol in general (especially for a uniform Clock service), but try to find out as much secret information as possible based on their inputs. More specifically, we assume cloud servers are more interested in file contents, changing time range in policy, and user access privilege than other secret information.
- Data users: Dishonest users would try to access files outside the scope of their access privileges. To achieve this, unauthorized users may intent to change the temporal constraints in his privilege independently or cooperatively. In addition, each party is preloaded with a private key and the public key can be easily obtained when necessary.

IV. Main Techniques

A. Main Idea

In order to achieve temporal access control on outsourced data in the cloud, we present and combine the following three advanced cryptographic techniques: integer comparison, current-time re-encryption and attribute-based encryption (ABE). The existing integer comparison scheme, first introduced in BSW's Cipher-policy ABE scheme, is a trivial method based on bitwise comparisons and AND/OR logical operations. Unfortunately, this method does not support the time attribute with range $A_k[t_a, t_b]$ in the private key SK, as well as the re-encryption mechanism. To resolve this challenging issue, we provide a new idea for designing cryptographic "one-way" property to represent the total ordering relation in integer. This means that given the integer relation $t_i \leq t_j$ and two corresponding value v_{t_i}, v_{t_j}, there exists an efficient algorithm to obtain v_{t_j} from v_{t_i}, but it is hard to compute v_{t_i} from v_{t_j}. Based on this idea, we have constructed a practical one-way function to realize the integer comparison. Also, we have demonstrated how to incorporate these functions into the BSW's scheme to realize fine-grained access control in clouds [10].

B. Forward/Backward Derivation Functions

Let time be denoted as a countable set $U = \{t_1, t_2, \cdots, t_T\}$ constituted from the discrete consecutive integers with total ordering $0 \leq t_1 \leq t_2 \leq \cdots \leq t_T \leq Z$, where Z is the maximum integer. In order to construct a cryptographic algorithm for integer comparison, we make use of a cryptographic map $\psi : U \rightarrow V$, where $V = \{v_{t_1}, \cdots, v_{t_T}\}$ is a set of cryptographic values. It is obvious that ψ must be an order-preserving map, that is a map such that if $t_i \leq t_j$ in U implies there exists a partial-order relation $\preceq$ to ensure $v_{t_i} \preceq v_{t_j}$ in V, where $v_{t_i} = \psi(t_i)$ and $v_{t_j} = \psi(t_j)$. In order to setup this kind of relation over V, we consider the partial-order relation in V as the "one-way" property in cryptography, as follows:

Definition 1: Given a function $f : V \rightarrow V$ based on a set $(U, \leq)$, it is called a forward derivation function if it satisfies the following conditions:

- ***Easy to compute**: the function f can be computed in a polynomial-time, if $t_i \leq t_j$, i.e., $v_{t_j} \leftarrow f_{t_i \leq t_j}(v_{t_i})$;*
- ***Hard to invert**: it is infeasible for any PPT algorithm to compute v_{t_i} from v_{t_j} if $t_i < t_j$.*

Similarly, we also define a function $\bar{f}$ to realize the derivation in opposite direction, which is called *Backward Derivation*. In order to avoid interference between f and $\bar{f}$, we use a different sign $\bar{\psi} : U \rightarrow \bar{V}$, and then define the backward derivation function $\bar{f} : \bar{V} \rightarrow \bar{V}$ based on the $\geq$ relation in $(U, \leq)$, e.g., $v_{t_j} \leftarrow f_{t_i \geq t_j}(v_{t_i})$.

C. Cryptographic Constructions

We propose a cryptographic construction for integer comparisons based on the forward/backward derivation functions. This construction is built on a special group $\mathbb{G}$ of RSA-type composite order $n = p'q'$. First, we choose two random secrets $\varphi, \bar{\varphi}$ in a group $\mathbb{G}$. Next, we choose two different random λ and μ in $\mathbb{Z}_n^*$, where the order of λ, μ are sufficiently large in $\mathbb{Z}_n^*$. Based on RSA system, we define two mapping functions $(\psi(\cdot), \bar{\psi}(\cdot))$ from an integer set $U = \{t_1, \cdots, t_T\}$ into $V = \{v_{t_1}, \cdots, v_{t_T}\}$ and $\bar{V} = \{\bar{v}_{t_1}, \cdots, \bar{v}_{t_T}\}$ as follows:

$$\begin{aligned} v_{t_i} &\leftarrow \psi(t_i) = \varphi^{\lambda^{t_i}} && \in \mathbb{G}; \\ \bar{v}_{t_i} &\leftarrow \bar{\psi}(t_i) = \bar{\varphi}^{\mu^{Z-t_i}} && \in \mathbb{G}. \end{aligned}$$

where, φ^{λ^t} denotes $\varphi^{(\lambda^t)}$ rather than $(\varphi^{\lambda})^t$. Note that, the values, $w_{t_i} = \lambda^{t_i}$ and $\bar{w}_{t_j} = u^{Z-t_j}$, can only be computed in the integer $\mathbb{Z}$ because n' and n are unknown based on the actual difficulty of factoring large numbers n. Next, according to the definition of $\psi(\cdot)$ and $\bar{\psi}(\cdot)$, it is easy to define the forward derivation function $f(\cdot)$ and backward derivation function $\bar{f}(\cdot)$ as

$$\begin{aligned} v_{t_j} &\leftarrow f_{t_i \leq t_j}(v_{t_i}) = (v_{t_i})^{\lambda^{t_j - t_i}} \in \mathbb{G}, \\ \bar{v}_{t_j} &\leftarrow \bar{f}_{t_i \geq t_j}(\bar{v}_{t_i}) = (\bar{v}_{t_i})^{\mu^{t_i - t_j}} \in \mathbb{G}. \end{aligned}$$

It is easy to show that $(\varphi^{\lambda^{t_i}})^{\lambda^{t_j - t_i}} = \varphi^{\lambda^{t_j}} = v_{t_j} \in \mathbb{G}$ and $(\bar{\varphi}^{\mu^{Z-t_i}})^{\mu^{t_i - t_j}} = \bar{\varphi}^{\mu^{Z-t_j}} = \bar{v}_{t_j} \in \mathbb{G}$. But it is intractable to obtain v_{t_i} from v_{t_j} for $t_i \leq t_j$ under the RSA assumption that λ^{-1} and μ^{-1} cannot be efficiently computed.

V. Security Analysis and Performance Evaluation

A. Security of Forward/Backward Derivation Functions

The security of TACE scheme is based on the RSA assumption and Gap Diffie-Hellman (GDH) assumption. Since this scheme is constructed based on BSW's CP-ABE scheme, it remains the security properties of their scheme, e.g., IND-CPA [1]. Hence, we focus on the security analysis of the different parts between them: we introduce the forward and backward derivation functions $f(\cdot)$, $\bar{f}(\cdot)$ into our scheme, so we need to assure the "one-way" property in the forward and backward derivation processes. This kind of "one-way" property can be guaranteed because the inverse of λ, u cannot be computed in $\mathbb{Z}_n^*$ if n is unknown. Thus, $\lambda^{t_c - t_i}, u^{t_j - t_c} \in \mathbb{Z}_n^*$ cannot be computed in $\mathbb{Z}$ for $t_c < t_i$ and $t_j < t_c$, so that $f_{t_c \leq t_i}(v_{t_i}) = (v_{t_i})^{\lambda^{t_c - t_i}}$ and $\bar{f}_{t_c \geq t_j}(v_{t_j}) = (\bar{v}_{t_j})^{\mu^{t_j - t_c}}$ is intractable. Strictly, this kind of "one-way" property can be proved under the RSA assumption: given an RSA public key (N, e) and a ciphertext $C = M^e \in \mathbb{G}$, it is infeasible to compute M.

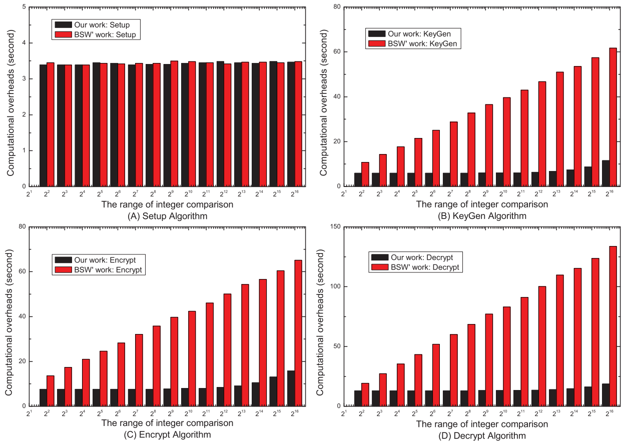
Theorem 1: Given a quintuple $(n, \lambda, t_i, \psi^{\lambda^{t_i}})$ over the RSA-type elliptic curve system $\mathbb{S}_N$, where ψ is unknown. It is infeasible to compute $(t_j, \psi^{\lambda^{t_j}})$ with $t_j < t_i$ for all probabilistic polynomial time (PPT) algorithms under the RSA assumption.

Proof: Seeking a contradiction, we assume that there exists a PPT algorithm $\mathcal{A}$ that can get a $(t_j, \psi^{\lambda^{t_j}})$ under above input, where $t_j < t_i$. This is equivalent to say that this algorithm can solve the RSA problem over elliptic curve for the public-key $(\mathbb{G}, N, e)$ and a ciphertext C, because the ciphertext can be computed by $M = R^{e^{t_i - t_j - 1}} \in \mathbb{G}$ if (t_j, R) is a solution of $\mathcal{A}$ on input $(n, \lambda = e, t_i, C)$ due to $R^{\lambda^{t_i - t_j}} = C = M^{\lambda}$, and $t_i - t_j - 1 \geq 0$. This contradicts the hypothesis. ■

B. Performance Evaluation

We have implemented our scheme in Qt/C++ and experiments were run on an Intel Core 2 processor with 2.16 GHz and 500M of RAM on Windows Server 2003. All disk operations were performed on a 1.82TB RAID 5 disk array. Using GMP and PBC libraries, we have implemented a cryptographic library upon which temporal attribute systems can be constructed.

We compare the performance of BSW's scheme and our scheme over integer ranges. We show the computational overheads for BSW's scheme and our scheme for different sizes of U in Figure 3. It is obvious that our scheme is more efficient than BSW's scheme. The reason is that the computation costs of algebraic operations and simple modular arithmetic operations can be neglected, because they run fast enough [11] in contrast with bilinear map operations. Without loss of generality, the performance of our scheme is better than that of BSW's scheme in $[1; 10,000,000]$.



(A) Setup Algorithm

(B) KeyGen Algorithm

(C) Encrypt Algorithm

(D) Decrypt Algorithm

Fig. 3. Computational overheads of BSW's scheme (Red) and our scheme (Black) for integer comparison operations: (a) Setup algorithm, (b) KegGen algorithm, (c) Encrypt algorithm, and (d) Decrypt algorithm.

Next, we analyze the storage and communication overheads of our TACE scheme. Thanks to the use of forward (or backward) derivation function for total ordering, TACE scheme has $O(1)$ size of private-key and ciphertext for a certain integer attribute in Table I, as well as a nearly linear-time complexity. But, for a comparison range $[1, Z]$, the storage and computation costs of BSW's scheme are nearly $O(\log_2 Z)$ times than those of our scheme. Hence, in comparison with BSW's scheme, TACE scheme provides a lower bound on variety of qualities, such as storage, communication and computation overheads.

TABLE I
COMPARISON OF BSW'S SCHEME AND OUR SCHEME.

	BSW's Scheme		Our Scheme	
	$t_1 \leq t$	$t \leq t_2$	$t_1 \leq t$	$t \leq t_2$
Ciphertext size	$\log_2 \lvert U\rvert$	$\log_2 \lvert U\rvert$	1	1
Private-key size	$\log_2 \lvert U\rvert$	$\log_2 \lvert U\rvert$	1	1
Depth of policy tree	$\log_2 \lvert U\rvert$	$\log_2 \lvert U\rvert$	1	1
Computation overhead	$\log_2 \lvert U\rvert$	$\log_2 \lvert U\rvert$	1	1

VI. RELATED WORK

In recent years, cryptographic access control [12], [13] has been introduced as a new access control paradigm to manage dynamic data sharing systems in cloud computing. It relies exclusively on cryptography to provide confidentiality of data managed by the systems, and is particularly designed to run in an untrusted or hostile environment which lacks of trust knowledge and global control [13]. Attribute-based encryption (ABE) is proposed to realize a fine-grained attribute-based access control mechanism. Since Sahai and Waters [14] introduced ABE as a new means for encrypted access control in 2005, ABE has received much attention and many schemes have been proposed in recent years, such as, key-policy ABE (KP-ABE) [4], [2] and ciphertext-policy ABE (CP-ABE) [1], [15]. For example, the model proposed by Yu *et al.* [3] introduced key-policy attribute-based encryption (KP-ABE) to achieve secure and scalable FGAC in cloud computing.

Temporal control is of particular significance and has been concerned in traditional access control [5], [16]. For example, in [5] the authors gave a temporal access control model and described applications in database systems and secure broadcasting. However, in the context of ABE, little work has been done on studying time control or integer comparison mechanisms. Even though Bethencourt *et al.* [1] gave a bitwise comparison method to realize integer comparison on CP-ABE scheme, it is unfortunately not efficient enough for practical applications.

VII. CONCLUSIONS

In this paper, we addressed the construction of temporal access control in cloud computing. Based on forward/backward derivation functions, we proposed a temporal access control encryption to support time range comparisons and re-encryption mechanism. We also discussed how to handle current time controls and temporal constraints with our solution.

REFERENCES

[1] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy*, 2007, pp. 321–334.

[2] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *ACM Conference on Computer and Communications Security*, 2007, pp. 195–203.

[3] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *INFOCOM, IEEE*, 2010, pp. 534–542.

[4] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *CCS, ACM*, 2006, pp. 89–98.

[5] E. Bertino, P. A. Bonatti, and E. Ferrari, "TRBAC: A temporal role-based access control model," *ACM Trans. Inf. Syst. Secur.*, vol. 4, no. 3, pp. 191–233, 2001.

[6] J. B. Joshi, E. Bertino, U. Latif, and A. Ghafoor, "A generalized temporal role-based access control model," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 4–23, 2005.

[7] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *CCS, ACM*, 2007, pp. 185–194.

[8] D. Boneh and X. Boyen, "Short signatures without random oracles," in *EUROCRYPT, Springer*, 2004, pp. 56–73.

[9] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Over-encryption: Management of access control evolution on outsourced data," in *VLDB, ACM*, 2007, pp. 123–134.

[10] Y. Zhu, H. Hu, G.-J. Ahn, M. Yu, and H. Zhao, "Comparison-based encryption for fine-grained access control in clouds," in *CODASPY, ACM*, 2012, To appear.

[11] P. S. L. M. Barreto, S. D. Galbraith, C. O'Eigeartaigh, and M. Scott, "Efficient pairing computation on supersingular abelian varieties," *Des. Codes Cryptography*, vol. 42, no. 3, pp. 239–271, 2007.

[12] A. Harrington and C. D. Jensen, "Cryptographic access control in a distributed file system," in *SACMAT, ACM*, 2003, pp. 158–165.

[13] A. V. D. M. Kayem, "Adaptive cryptographic access control for dynamic data sharing environments," Ph.D Thesis, Queens University Kingston, Ontario, Canada, October 2008.

[14] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT, Springer*, 2005, pp. 457–473.

[15] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *ICALP (2)*, 2008, pp. 579–591.

[16] E. Bertino, B. Carminati, and E. Ferrari, "A temporal key management scheme for secure broadcasting of xml documents," in *CCS, ACM*, 2002, pp. 31–40.