A Multilevel Access Control Scheme for Data Security in Transparent Computing

Tao Peng[1], Qin Liu[2], Guojun Wang[3,1,*]

[1.] School of Information Science and Engineering, Central South University
Changsha, Hunan Province, P. R. China, 410083
Email: pengtao_work@163.com

[2.] School of Information Science and Engineering, Hunan University
Changsha, Hunan Province, P. R. China, 410082
Email: gracelq628@hnu.edu.cn

[3.] School of Computer Science and Educational Software, Guangzhou University
Guangzhou, Guangdong Province, P. R. China, 510006
*Correspondence to: csgjwang@gmail.com

Abstract—In transparent computing, the client terminals are rather light-weighted, while all of the resources (including the operating systems, OSs for short) are stored on remote servers, and delivered on demand to clients in a streaming way. In this paper, we propose a Multilevel Access Control Scheme in Transparent Computing (MACTC) to protect user data with different security levels, and provide multilevel access control and valid identity authentication. The proposed scheme is effective in multilevel data security, flexible in authorized resource sharing, and secure against various malicious attacks. Experiment results verify the feasibility of our scheme.

Keywords—Transparent computing; Computer security; Authentication; Privacy; Multilevel security; Access control

I. INTRODUCTION

Over the past years, computing paradigms have greatly evolved with the rapid development of computer network and information technologies. Transparent computing [1][2][3] is one of the emerging technologies, which allows users to enjoy user-controlled services by extending the stored program concept in the von Neumann architecture into the networking environments spatio-temporally. Transparent computing loads a variety of heterogeneous OSs and applications dynamically on different device. This feature enables users to focus on the available application services without caring about which physical device will be used and what OS should be run on it.

The new mechanism comes with many advantages in information security aspect [4][5]. The centralized management at servers can bring convenience to the protection of users data, and reduce the risks of information leakage and data theft. However, this uniqueness has brought new challenges in service reliability and security, since the OSs, applications and data are centralized in servers, and they are shared by all users in transparent computing system. We envision such a scenario: An enterprise introduces the transparent computing as its office system, due to the desired features

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

of transparent computing. Some information, (e.g., files, tables, data, etc.) will be produced during the day-to-day work of the employees in this enterprise. The produced data has different security levels and access permissions. For example, the open files can be shared with everyone, but some sensitive tables may be revealed to specific users, and other private personal information can not be disclosed to anyone. Thus, according to their sensitivity, users classify the information into three categories: public information, sensitive information, private information. While users in transparent computing are supposed to reserve no storage space on their clients, all execution results and data must be stored to the Transparent Servers (TSs). Without users' consent, the data stored in servers may be abused or misused by unauthorized accesses or server managers. Therefore, a secure protection scheme is imperative to encrypt the private information of each user before storing data into TSs, and the scheme is supposed to protect user information with multilevel security, and provide precise access control to them as well. Some existing multiple-receiver encryption schemes use Attribute-Based Encryption (ABE) [6][7] to achieve multilevel confidentiality and finegrained access control, but these methods consume large computation cost due to the bilinear map operations during encryption and decryption. Moreover, effective user revocation is an intractable issue in these schemes, since the data should be re-encrypted when privilege is revoked. How to protect multilevel data security and achieve authorized resource sharing in an efficient and flexible way in such an environment has become a problem. In this paper, we propose a Multilevel Access Control Scheme in Transparent Computing (MACTC) to protect user data with different security levels. The proposed scheme introduces an Authentication Server (AS), which acts as "Authentication Authority", to perform multilevel access control and identity authentication, dealing with user data access, storage, transmission, and processing in transparent computing environment. The scheme is essentially based on the following attractive characteristics and capabilities of transparent computing.

1) We are among the first to consider the problem of multilevel security of user data in transparent computing environment. We design a multilevel access control scheme, which enables a privilege user to access the specified files under the verification of different level access control polynomials.

2) Our scheme has an overall consideration of multilevel data security, effective access control, and user identity authentication for integrated technologies to provide a security structure in transparent computing.

3) We use selective multi-modality biometric method to validate users' claimed identity, by which user can choose the biometric input modality according to their devices and environment. Beyond the traditional model, it can be applied to cross-platform, and thus it is more suitable for transparent computing applications.

II. SYSTEM MODEL

MACTC mainly has three parties: the user/TC (Transparent Client), the AS, and the TS, the frame structure of proposed scheme is shown in Fig. 1.

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

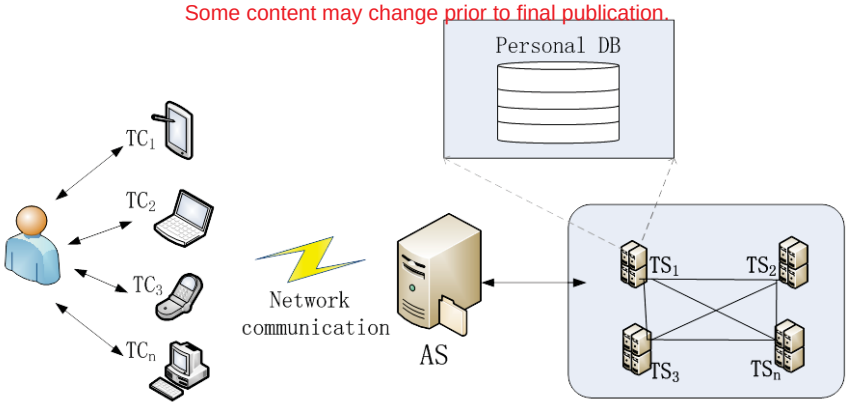

Fig. 1. System configuration of scheme

In our scheme, we regard a user and a TC as one party after the successful verification between them. We introduce an AS, a Third Trust Party (TTP)-based entity, into the scheme, which is located in front of TSs. The task of AS is to authenticate a legitimate user and to verify his read and write permissions to protected data that the user is interested in. We assume that the AS is deployed in a small and medium business that has implemented the general work in transparent computing environment. For ease of explanation, in this paper, we only use a single AS, but multiple ASs can be deployed as necessary.

In consideration to the diversification of users' demands in transparent computing environment (user only needs basic username/password authentication while using a personal desktop, but the users may require different biometric information for enhanced security while using mobile devices), we use selective multi-modality biometric strategy to validate the individual's identity, including fingerprint, palmprint, voice, image, and so on. Users can choose the biometric input modality according to their hardware and software platforms and environments to perform the identity authentication.

III.TECHNIQUE PRELIMINARIES

A. Transparent computing

Transparent computing is aimed at providing cross-platform experience for users transparently and seamlessly. In transparent computing, all of the resources (including OSs) are stored on remote Transparent Servers, while the Transparent Clients, are rather light-weighted and almost like a "bare" computer. Managed by the transparent operating system, META OS [8], the instance OS and other resources can be delivered through the network and be requested on demand for local execution in a block-streaming way. The Transparent computing paradigm can achieve the following desired features[2][3]:

1) The use of transparent computing technologies is transparent to users and applications, and requires no application modification. The installation, maintenance and management of service and resources will be carried out by the system itself or centrally on servers by administrators.

2) Users can flexibly choose any needed OS, toolkits and applications from the remote servers, running on various types of hardware platforms.

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

3) The software and data are dynamically scheduled to clients in a streaming way.

B. Information classification and multilevel database

According to the sensitivity, users classify information into three categories, A, B, C, and the classification is as follows: A level is public information, which is shared with all legal users, so anyone in the system can access A level of data. B level is sensitive information, partly public, which is shared with the authorized users, such as the authorized colleague and team members, this B level of information requires to be encrypted with distinctive encryption keys. C level is private information, not public, which cannot be shared with anyone, this C level of information can be conducted encryption and decryption by the user himself, and the encryption keys should not be revealed to anyone. The classifying standard is determined by a user with his subjective standard. Users can set up their routine daily information with the fixed level in advance when they enroll in transparent computing office system, while other information is optional. In this way, users can control their own information by themselves.

C. Polynomial for multilevel database access control of the protected data

We use the polynomial generated by the AS to verify user's privilege to access each file of corresponding security level in the database. Similar to the work in [9], we propose the access control polynomials for our MACTC scheme from the following 2 parts:

1. Polynomial for multilevel database access control

$$LP_i = r_{i1} + (x - B_i)\{r_{i2} + (x - C_i)\} \quad (1)$$

$$B_i = h(b_i), C_i = h(c_i) + h(BT_i) \quad (2)$$

$$BL_i = LP_i(B_i) = r_{i1} \quad (3)$$

$$CL_i = LP_i(C_i) = r_{i1} + r_{i2}(C_i - B_i) \quad (4)$$

where r_{i1}, r_{i2} are random numbers, BL_i ,CL_i are access control polynomials generated by the AS to control that a user i access to B and C level of data. We define B_i , C_i as "Level Authentication Identifier", which means an accessible level of a user i. With the "Level Authentication Value", b_i , c_i got from the AS, a user i has the chance to obtain correct B_i , C_i using Eq. 2 to verify his validity and get the privileges to B and C level of data. Especially, in order to retrieve the private C level of data, a user has to provide his biometric, the TC captures template information and computes hash value of it, $h(BT_i)$. Only when the user provides a biometric whose hash value matches the one stored on the AS, he can pass the validation process. In this way, the scheme can guarantee high security of the C level of data.

2. Polynomial for file-level access control of sharable data

For each file in B level of data for user i, defined as BF_{ij} , user can designate a set

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

of l users to share this file. The AS will generate the following polynomial for file-level access control, when each file is successfully saved to the personal database on the TS.

$$FP_{ij} = r_{i1}(x - u_1\_BF_{ij})(x - u_2\_BF_{ij})...(x - u_i\_BF_{ij})...(x - u_l\_BF_{ij}) + r_{i2} \quad (5)$$

where r_{i1}, r_{i2} are random values, i, j ($j < m$, m is the total number of B level of files) and l ($l < n$, n is the total number of users) are positive integers. The polynomial FP_{ij} is l-degree polynomial, it controls the set of users who can access to the jth file for user i's B level of information. We define $u_l\_BF_{ij}$ as "File Authentication Value", which user l can get from the AS to access the jth file in B level data of a user i. Especially, for all $1 \le j \le m$, we set $u_i\_BF_{ij} = b_i$, which means the user i can access all his own files in B level of data, just by providing the "Level Authentication Value", b_i.

IV. PROPOSED SCHEME

In this section, we present the MACTC scheme, which includes the registration phase, login phase, multilevel access control phase and password or authorization change phase.

A. Registration phase

The process of registration phase is as follows:

R1: The user i chooses his identity (ID_i), password (PW_i) for registration, and entries his biometric (including image, fingerprint, palmprint information, denoted as (I_i, F_i, P_i) into the scanner embedded device or record his voice (denoted as V_i) on the recording equipment in TC, which captures and stores the biometric template as BT_i.

R2: User i encrypts the information with public key of the AS, pk_{as}, which is $E^*_{pk_{as}}(ID_i, h(PW_i), h(BT_i))$, where $h(\cdot)$ is collision resistant hash function.

R3: i sends registration information to the AS. The message from user to the AS is:

$$MSG_{U2A} = \{E^*_{pk_{as}}(ID_i, h(PW_i), h(BT_i))\}$$

R4: The AS decrypts this message with its private key, then checks its user list and confirms the validity of the registration request. If i is a fresh registrant, the AS encrypts user's registration as $E_{k_{as}}(h(PW_i), h(BT_i))$, and then creates a new entry, stores the registration information of i. Otherwise, ID_i exists in user list, the AS returns the information "ID_i already exists" to user i.

B. Login phase

After user i registers to the AS, when i wants to log into the server, the login and authentication phase works as follows:

L1: User i chooses the biometric modality he wants to use according to his TC's

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

hardware configuration and environment, and entries one of the biometric information, such as fingerprint, palmprint voice, image and so on, the TC captures the template information as $I_i^*, F_i^*, P_i^*, V_i^*$ respectively.

L2: User *i* inputs his password PW_i^*, and biometric BT_i^*. TC verifies them with stored ones: $h(PW_i) = ?h(PW_i^*), h(BT)_i = ?h(BT_i^*)$. If it is not correct, *i* reenters.

L3: User *i* randomly generates a value, α, with the same size as hash value's output, which will be used as a session key and a masking value. The random value should be generated newly every session and should be different every time. Then TC computes

$$a = E_{pk_{as}}^*(t), b = h(PW_i^*)\alpha + t, c = h(BT_i^*) + \alpha$$

and $d = h(a|b|c)$, where pk_{as} is the public key of the AS. In the expression of *b*, we add the timestamp, *t*, to the value of $h(PW_i^*)\alpha$. The reason is that $h(PW_i^*)$ is a constant parameter, if $b = h(PW_i^*)\alpha$, an attacker may obtain the common factor $h(PW_i^*) = \frac{b}{\alpha}$ from successive attacks to *b*.

L4: User *i* sends his login request and related information to the AS. The message from user to the AS is:

$$MSG_{U2A} = \{a, b, c, d\}$$

L5: The AS performs integrity check of the message by $d = ?h(a|b|c)$

L6: Upon receiving the massage from the user *i*, the AS decrypts $a = E_{pk_{as}}^*(t)$ with its private key sk_{as}, and checks $t \in TTL$ (Time To Live).

L7: If all verifications are successful, the AS decrypts the user *i*'s registration information $E_{k_{as}}(h(PW_i), h(BT_i))$, which is stored in the AS' database.

L8: The AS computes the value of α by $\alpha = \frac{b-t}{h(PW_i)}$, where $h(PW_i)$ is the decrypted value from point L7. As long as $h(PW_i^*) = h(PW_i)$, the AS can get the correct α. With this α, the AS computes $h(BT_i^*)$ by $h(BT_i^*) = c - a$. Then the AS checks if the $h(BT_i^*)$ matches the biometric template in its database.

L9: The user *i* logins successfully, if he can pass all the above validation steps.

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

C. Multilevel access control phase

We introduce the multilevel access control from the processes of writing, reading and updating files as follows:

Write files :

MW1: If a user *i* tries to save his data into the remote TS, referring to B and C level of files, he should encrypt them on the TC, before uploading the information.

$$E_{k\_BF_{ij}}(BF_{ij}); E_{k\_CF_{ij}}(CF_{ij})$$

where the BF_{ij} ,CF_{ij} are the *j*th file of *i*'s B/C level of data, k_BF_{ij} and k_CF_{ij} are the secret keys of them. Hereafter, we assume $x = E_{k\_BF_{ij}}(BF_{ij}); y = E_{k\_CF_{ij}}(CF_{ij})$

MW2: User *i* names the specific users to share this B level of data. The set of privileged user can be represented by $U_i = u_1, u_2 ... u_l$

MW3: User *i* encrypts the requirement using session key, α , which is generated during the login phase, and sends message to the AS:

$$MSG_{U2A} = \{E_\alpha(w\_B_{ij}, x, U_i; w\_C_{ij}, y)\}$$

MW4: The AS decrypts message with α , implements *i*'s demand, and then transfers the queries to corresponding TS in the form of $E_{k_{at}}(x, y)$, where k_{at} is a secret key shared between the AS and the TS. The message from the AS to TS is:

$$MSG_{A2T} = \{E_{k_{at}}(x, y)\}$$

MW5: Upon receiving the message from the AS, the TS decrypts it with k_{at}, and executes the task of writing files in the user *i*'s database. During this process, the data exists in the form of encryption, neither the AS nor the TS can read the plaintext file.

MW6: Once the files are successfully saved on database of TS, the AS will generate multilevel database access control polynomial LP_i by Eq.1, and file-level database access control polynomial FP_{ij} by Eq.5.

MW7: The AS returns Level Authentication Values, b_i and c_i, to user *i*, and distributes the File Authentication Values, $\{u_l\_BF_{ij}\}$ to users $u_1, u_2 ... u_l$ With which user *l* can access the *j*th file of B level of data from a user *i*'s DB. The message from the AS to user is:

$$MSG_{A2U} = \{E_\alpha(b_i, c_i)\}$$

Read files:

A user can access his own personal DB with the Level Authentication Values, and access other's personal DB with the File Authentication Values. We will take an example to illustrate the process of reading files. A user *i* tries to retrieve the *j*th file of B and C level of information from his own DB and other user *o*'s B level of data in *o*'s personal DB.

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

MR1: Using the session key, α, user i encrypts query message, it includes the Level Authentication Values, b_i , c_i, and the File Authentication Value $u_i\_BF_{oj}$, which i can get from the AS, once i was appointed as one of the privileged users for the jth file of o's B level of data. Referring the C level of data, i should also provide one of the biometric information, $h(BT_i^*)$. Then i sends message to the AS:

$$MSG_{U2A} = \{E_\alpha(r\_B_{ij}, b_i; r\_C_{ij}, c_i, h(BT_i^*); r\_B_{oj}, u_i\_BF_{oj})\}$$

MR2: The AS decrypts message, then computes the Level Authentication Identifiers, B_i , C_i, with the submitted b_i , c_i and $h(BT_i^*)$

$$B_i = h(b_i), C_i = h(c_i) + h(BT_i^*)$$

MR3: The AS checks $LP_i(B_i) = ?BL_i$, and $LP_i(C_i) = ?CL_i$, where BL_i ,CL_i are generated during write file phase in MW5. This access control process is to verify a user i has permission to retrieve the corresponding level of data in his own personal DB or not.

MR4: In this step, the AS checks user i's ability to access other's personal DB. The AS first checks if $ID_i \in U_o$, where U_o is the authorized user set assigned by user o during his write file phase in MW2. Then AS computes the polynomial for file-level access control and checks $FP_{oj}(u_i\_BF_{oj}) = ?FP_{oj}$, where FP_{oj} is generated when user o saved successfully his jth file of B level data in personal DB.

MR5: If i is conformed to all the access control rules, the AS queries TS with $E_{k_{at}}(r\_B_{ij}, r\_C_{ij}, r\_B_{oj})$, where k_{at} is a secret key shared between the AS and the TS.

MR6: TS decrypts message with k_{at}, conducts AS' query and returns results to AS with s, x, y, z ,where $s = h(x \mid y \mid z)$, $x = E_{k\_BF_{ij}}(BF_{ij})$; $y = E_{k\_CF_{ij}}(CF_{ij})$; $z = E_{k\_BF_{oj}}(BF_{oj})$. The message from TS to the AS is:

$$MSG_{T2A} = \{x, y, z, s\}$$

MR7: The AS checks integrity of information by $s = ?h(x \mid y \mid z)$, then encrypts the results with session key, and transfers them to i. The message from the AS to user is: $MSG_{A2U} = \{E_\alpha(x, y, z)\}$

MR8: User i decrypts the message with α, and gets the origin data with the encryption keys, $k\_BF_{ij}, k\_CF_{ij}, k\_BF_{oj}$, where the key k_BF_{oj} can get by negotiating

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

with user *o*. Finally, user *i* can obtain what he wants as follows:

$$D_{k\_BF_{ij}}(E_{k\_BF_{ij}}(x)) = BF_{ij},\ C_{k\_CF_{ij}}(E_{k\_CF_{ij}}(y)) = CF_{ij}, D_{k\_BF_{oj}}(E_{k\_BF_{oj}}(z)) = BF_{oj}$$

Update files

If the user wants to update his existing data in his own personal database, involving the B and C level of information, he must first pass the certification on the AS to get the permission of updating files.

MU1: User *i* sends his request and related authentication values to the AS. Before submitting the message, he should encrypts it with session key. The message from user to the AS is: $MSG_{U2A} = \{E_{\alpha}(q)\}$, where

$$q = \{u\_B_{ij}, E_{k\_BF_{ij}}(uBF_{ij}), b_i; u\_C_{ij}, E_{k\_CF_{ij}}(uCF_{ij}), c_i, h(BT_i^*)\}$$

uBF_{ij} and uCF_{ij} are the updated files of *i*.

MU2: The AS decrypts message, then computes the Level Authentication Identifiers, B_i , C_i, with the submitted b_i, c_i and $h(BT_i^*)$.

$$B_i = h(b_i), C_i = h(c_i) + h(BT_i^*)$$

MU3: The AS checks $LP_i(B_i) = ?BL_i$, and $LP_i(C_i) = ?CL_i$, where BL_i ,CL_i are generated during write file phase in MW5. This process is to verify whether a user *i* is legitimate to update the corresponding level of data in personal DB or not.

MU4: If the authentication is successful, the AS transfers the encrypted latest information with k_{at} to TS. We set $x' = E_{k\_BF_{ij}}(uBF_{ij}); y' = E_{k\_CF_{ij}}(uCF_{ij})$. The message from the AS to TS is:

$$MSG_{A2T} = \{E_{k_{at}}(x', y')\}$$

MU5: Upon receiving the message from the AS, the TS decrypts it, and performs the procedure of updating files.

D. Password or authorization change phase

Password change

P1: User *i* presents password change request to the registration server AS, after *i* logins the system, he has to provide original password PW_i and the biometric information to authenticate on the AS.

$$MSG_{U2A} = \{E_{\alpha}(h(PW_i^{'}), h(PW_i^*), h(BT_i^*))\}$$

P2: The AS decrypts stored registration information in his database, and checks the hash value of password and biometric with the stored one.

P3: If verification results are correct, the AS conducts the update of *i*'s password, and encrypts the latest information with his private key, then replaced the one in his database.

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

Authorized user change

When user *i* wants to revoke someone's privilege, or to change the set of authorized user for his B level of information *j*th file, *i* should negotiate with the AS.

A1: User *i* presents the authorized user modification request for his B level of file, after *i* logins the system, and he also should provide the Level Authentication Value, b_i, to pass the verification procedure on the AS.

A2: The AS validates the user *i*, and checks $LP_i(b_i) = ? BL_i$.

A3: If the verification is successful, the AS updates the set of authorized user, U_i, and changes the corresponding file-level access control polynomial, FP_{ij} for user *i*'s *j*th file. The disqualified users cannot access this file any more since the authorized user set and polynomial have been changed.

V. PERFORMANCE ANALYSIS AND EVALUATION

In this section, we will provide the performance analysis and evaluation of the proposed scheme.

A. Performance analysis

There are four phases in our scheme, we mainly analyze the performance of login phase and access control phase, since these two phase are principal parts of the proposed protocol and should be implemented for each session. We mainly analyze the performance of MACTC protocol from the angles of the TC side, the AS and the TS. We believe that the most expensive computation is the asymmetric encryption $E^*(K, X)$ and asymmetric decryption $D^*(K, Y)$, abbreviated as E^* and D^*, the next is the symmetric encryption $E(K, X)$ and the symmetric encryption $D(K, Y)$, abbreviated as E and D; and then, one-way hash function, $h(\cdot)$, abbreviated as H. In the following analysis, we focus on above three operations, since the other operations, such as modular multiplication, concatenation, XOR operation, etc., require very few computations, their computation cost is neglected here. Especially, the biometric authentication process is much more affected by environmental factors (e.g., the diverse TCs and extraction methods or matching algorithms, etc.) other than by the scheme we designed newly. We did not develop a new biometric authentication algorithm but use known efficient method to proposed scheme. Thus, we exclude this part from our performance analysis.

During login phase, TC should submit the encrypted timestamp, the hashed user's password and biometric. From the AS side, it first needs to conduct integrity check of the message, decrypts the timestamp to check TTL, and then decrypts stored registration information of user to perform identity authentication. Hence, the proposed protocol needs $3H + E^*$ computations on the user side and $H + D^* + D$ computations on the AS during the login stage.

During access control phase, performance is up to the attribute and amount of data

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

which a user wants to store or to retrieve. We take the example that a user tries to write, read, and update respectively B level of file on TSs to illustrate the computation performance. This level of information should be encryption or decryption, when uploading or downloading from TSs. Messages between TC and the AS are encrypted by an session key, and messages between the AS and TSs are encrypted by the shared key. When getting results from TSs, the AS and TS should check integrity of messages. Especially, in this phase, the AS needs to generate or verify LP_i and FP_{ij} to conduct access control to B and C level of file. The LP_i is three degree polynomial, whose computation time is relatively low. The FP_{ij} is l degree polynomial, if the number of authorized users is l. We can build FP_{ij} based on the Eq.5, and set each factor of the polynomial as distinct floating point number. The running time of generating and calculating these polynomials is ignorable. Therefore, the computation overhead on the TC, AS and TS is listed in Table I.

TABLE I. THE PERFORMANCE OF LOGIN AND ACCESS CONTROL PHASE

	Login	Access control		
		Write	Read	Update
TC	*E*+3H*	*2E+D*	*E+D*	*2E*
AS	*D*+D+H*	*3E+D*	*2E+D+H*	*E+D*
TS	*0*	*D*	*D+H*	*D*

The communication cost of authentication includes cost of transmitting messages involved in the proposed scheme. In our scheme it is to a large extent relies on the information which users want to access. For all implementations throughout four phases in MACTC scheme, the authentication and access control methods such as password, biometric, audiovisual, and access control polynomials are applied by only one round protocol. If one of these verification processes would fail, the protocol would be terminated. This is the minimum number of rounds to achieve authentication, data sharing or access control processes, and all of the other functions. Therefore, we can say that communication overheads for the TC, the AS and the TS are reasonable in the proposed scheme.

B. Evaluation

We evaluate the performance of MACTC scheme in terms of running time for various operations in that phase. Our experiments are implemented with Java Development Kit (JDK)-1.7 and Eclipse Integrated Development Environment (IDE), running on a local machine with an Intel Core-i5 2.5GHz, 2GB RAM and Window7 OS. The performance of involved phases depends on the size of messages and files. The size of messages are set to 32, 64, 128, 512, 1024 Bytes, and the size of files are set to 1, 100, 500, 1000, 5000 *K*Bytes. We use the asymmetric encryption algorithm, RSA, to treat short messages (e.g., timestamp). The symmetric encryption algorithm, 256-bit-AES, is used to encrypt messages and files. Hash function, 256-bit-SHA, is used to ensure the integrity of data. The computation cost of them are shown in the Table II.

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

TABLE II. THE COMPUTATION COST

	Messages(Bytes)					Files(*K*Bytes)				
	32	64	128	512	1024	1	100	500	1000	5000
$E^*(ms)$	117	117	118	120	125	/	/	/	/	/
$D^*(ms)$	7	17	20	50	93	/	/	/	/	/
E(ms)	83	85	88	93	98	97	147	253	386	1482
D(ms)	0.8	0.8	1	2	3	3	28	92	176	862
H(ms)	7	8	9	11	16	13	17	21	30	101

From the evaluation, we can find that the most expensive operation is asymmetric encryption, it will take about 0.1s-0.2s to encrypt a short message, but Table I shows that this operation needs to be performed only once during login phase. The running time of hash function (256-bit-SHA) is no more than 0.02s for messages, and is about 0.03s for a 1000*K*Bytes file. The symmetric encryption and decryption (256-bit-AES) consume main computation time. It takes 0.1s encryption time and 0.003s decryption time for a 1*K*Bytes message, and takes 0.4s encryption time and 0.2s decryption time for a 1000*K*Bytes file. From the user side, the file level of encryption and decryption are performed once or twice in each session (depends on the different operations of the user), we can say the computation overheads of our scheme are reasonable, since the server is regarded as a powerful device.

VI. CONCLUSION

In this paper, we proposed a comprehensive MACTC scheme for protection of user data with multilevel security. Our goal is to provide security management for the user data access, storage, transmission, and processing in transparent computing. In our future work, we will improve our scheme by deploying multiple ASs to avoid the potential bottleneck between the users and the TSs, and ensure the high availability of the system.

ACKNOWLEDGMENT

This research was supported in part by the National Natural Science Foundation of China under grant numbers 61472451&61272151&61402161, the International S&T Cooperation Program of China under grant 2013DFB10070, the Hunan Provincial Science Technology Program of China under grant 2012GK4106, and the Hunan Provincial Natural Science Foundation of China under grant 2015JJ3046.

REFERENCES

[1]. Y. Zhang and Y. Zhou, “Transparent Computing: Spatio-temporal Extension on Von Neumann Architecture for Cloud Services,” *Tsinghua Science and Technology*, vol. 18, no. 1, 2013, pp. 10–21.

[2]. Y. Zhang and Y. Zhou, “Transparent Computing: a New Paradigm for Pervasive

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

Computing," *Ubiquitous Intelligence and Computing:2006 International Conf.* (UIC 06), 2006, pp. 1–11.

[3]. Y. Zhang and Y. Zhou, "TransOS: a Transparent Computing-based Operating System for the Cloud," *International Journal of Cloud Computing*, vol. 1, no. 4, 2012, pp. 287–301.

[4]. Y. Zhang, L. T. Yang, Y. Zhou, and W. Kuang, "Information Security Underlying Transparent Computing: Impacts, Visions and Challenges," *Web Intelligence and Agent Systems*, vol. 8, no. 2, 2010, pp. 203–217.

[5]. G. Wang, Q. Liu, Y. Xiang, and J. Chen, "Security from the Transparent Computing Aspect," *Pro. 2014 IEEE Conf. Computing, Networking and Communications* (ICNC), 2014, pp. 216–220.

[6]. Q. Liu, G. Wang, and J. Wu, "Time-based Proxy Re-encryption Scheme for Secure Data Sharing in a Cloud Environment," *Information Sciences*, vol. 258, 2014, pp. 355–370.

[7]. G. Wang, Q. Liu, J. Wu, and M. Guo, "Hierarchical Attribute-based Encryption and Scalable User Revocation for Sharing Data in Cloud Servers," *Computers & Security*, vol. 30, no. 5, 2011, pp. 320–331.

[8]. Y. Zhang and Y. Zhou, "4VP: a Novel Meta OS Approach for Streaming Programs in Ubiquitous Computing," *Advanced Information Networking and Applications: 21st International Conf.* (AINA 07), 2007, pp. 394– 403.

[9]. H.-A. Park, J. W. Hong, J. H. Park, J. Zhan, and D. H. Lee, "Combined Authentication-based Multilevel Access Control in Mobile Application for Dailylifeservice," *IEEE Transactions on Mobile Computing*, vol. 9, no. 6, 2010, pp. 824–837.

Tao Peng received the B.Sc. degree in computer science from Xiangtan University, Xiangtan, China, in 2004 and the M.Sc. degree in circuits and systems from Hunan Normal University, Changsha, China, in 2007. She is currently working toward the Ph.D. degree in the School of Information Science and Engineering, Central South University, Changsha, China. Her research interests include network and information security issues.

Qin Liu received her B.Sc. in Computer Science in 2004 from Hunan Normal University, China, received her M.Sc. in Computer Science in 2007, and received her Ph.D. in Computer Science in 2012 from Central South University, China. She has been a Visiting Student at Temple University, USA. Her research interests include security and privacy issues in cloud computing. She is an Assistant Professor in the College of Computer Science and

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE

Electronic Engineering at Hunan University, China.

Guojun Wang received his B.Sc. in Geophysics, in 1992, M.Sc. in Computer Science, in 1996, and Ph.D. in Computer Science, in 2002, from Central South University, China. He is Professor of both Guangzhou University and Central South University. He has been an Adjunct Professor at Temple University, USA; a Visiting Scholar at Florida Atlantic University, USA; a Visiting Researcher at the University of Aizu, Japan; and a Research Fellow at the Hong Kong Polytechnic University. His research interests include network and information security, Internet of things, and cloud computing. He is a distinguished member of CCF, and a member of IEEE, ACM, and IEICE.

Digital Object Identifier 10.1109/MCSE.2016.50 1070-9924/$26.00 2016 IEEE