Identity-Based Cryptography for Cloud Security

Hongwei Li[1], Yuanshun Dai[1,2] , Bo Yang[1]

[1] University of Electronic Science and Technology of China
hongwei.uestc@gmail.com
yangbo_cd@126.com
[2] University of Tennessee, Knoxville, USA.
ydai1@eecs.utk.edu

***Abstract*—Cloud computing is a style of computing in which dynamically scalable and commonly virtualized resources are provided as a service over the Internet. This paper, first presents a novel Hierarchical Architecture for Cloud Computing (HACC). Then, Identity-Based Encryption (IBE) and Identity-Based Signature (IBS) for HACC are proposed. Finally, an Authentication Protocol for Cloud Computing (APCC) is presented. Performance analysis indicates that APCC is more efficient and lightweight than SSL Authentication Protocol (SAP), especially for the user side. This aligns well with the idea of cloud computing to allow the users with a platform of limited performance to outsource their computational tasks to more powerful servers.**

***Index Terms*—cloud computing, identity-based cryptography, encryption, signature, authentication.**

ACRONYM

APCC	Authentication Protocol for Cloud Computing
HACC	Hierarchical Architecture for Cloud Computing
HIBE	Hierarchical Identity-Based Encryption
IBC	Identity-Based Cryptography
IBE	Identity-Based Encryption
IBS	Identity-Based Signature
PaaS	Platform as a Service
PKG	Private Key Generator
SaaS	Software as a Service
SSL	Secure Sockets Layer
SAP	SSL Authentication Protocol
TLS	Transport Layer Security

NOTATION

C	Client
S	Server
n_C, n_S	The fresh random number
SID	The session identifier
$specification_C$	The cipher specification of Client C
$specification_S$	The cipher specification of Server S
S_{CS}	A pre-master secret used to generate the shared key
$E_{P_S}[S_{CS}]$	Encrypt S_{CS} with the public key P_S of Server S using the encryption algorithm of IBE
M	All handshake messages since the ClientHello message
$Sig_{S_C}[M]$	Sign M with the private key S_C of C using the signature algorithm of IBS
$Ver_{ID_C}(Sig_{S_C}[M])$	Verify the signature $Sig_{S_C}[M]$ with the help of ID_C using the verification algorithm of IBS
$D_{S_S}(E_{P_S}[S_{CS}])$	Decrypt the $E_{P_S}[S_{CS}]$ with the private key S_S of Server S using the decryption algorithm of IBE

I. INTRODUCTION

Cloud computing is a class of the next generation highly scalable distributed computing platform in which computing resources are offered 'as a service' leveraging virtualization and Internet technologies[1] [2]. Cloud-based services include Software-as-a-Service (SaaS) and Platform as a Service (PaaS). Amazon's Elastic Compute Cloud (EC2) [3] and IBM's Blue Cloud [4] are examples of cloud computing services. These cloud service providers allow users to instantiate cloud services on demand and thus purchase precisely the capacity they require when they require based on pay-per-use or subscription-based model.

Although cloud computing provides a number of advantages that include economies of scale, dynamic provisioning, increased flexibility and low capital expenditures, it also introduces a range of new security risks [5]. As cloud computing brings with it new deployment and associated adversarial models and vulnerabilities, it is imperative that security takes center stage. This is especially true as cloud computing services are being used for e-commerce applications, medical record services, and back-office business applications, all of which require strong confidentiality guarantees. Thus, to take full advantage of the power of cloud computing, end users need comprehensive security solutions to attain assurance of the cloud's treatment of security issues.

Independent of cloud computing, a variant of traditional public key technologies called Identity-Based Cryptography (IBC) [6, 7] has recently received considerable attention. Through IBC, an identifier which represents a user can be transformed into his public key and used on-the-fly without any authenticity check. The potential of IBC to provide greater flexibility to entities within a security infrastructure and its certificate-free approach may well match the dynamic qualities of cloud environment. In other words, it seems that the development of IBC may offer more lightweight and flexible key usage and management approaches within cloud security infrastructures than traditional PKI does. The application of IBC in cloud computing is an emerging and interesting area.

Our contributions. In this paper, we would like to examine what can be achieved in a fully identity-based approach for cloud environment. Specifically, our main contributions include:

1. We propose a Hierarchical Architecture for Cloud Computing (HACC). It inherits attractive properties from IBC such as being certificate-free and having small key sizes. This potentially offers a more lightweight key management approach.
2. Based on the Hierarchical Architecture for Cloud Computing (HACC), we present Identity-Based Encryption (IBE) and Identity-Based Signature (IBS) for cloud computing.
3. Based on the above 1 and 2, we design an Authentication Protocol for Cloud Computing (APCC). APCC is more efficient and lightweight than SSL Authentication Protocol (SAP) [8], especially for the user side, which contributes good scalability to the much larger cloud systems.

Organization. The remainder of the paper is organized as follows. Section Ⅱ introduces related work. Section Ⅲ presents some preliminaries. In section Ⅳ, we propose the Hierarchical Architecture for Cloud Computing (HACC). Section Ⅴ describes the identity-based encryption and signature for the HACC. Section Ⅵ presents a secure authentication protocol for cloud computing. Section Ⅶ makes the performance analysis for our new protocol. Section Ⅷ illustrates some simulations to evaluate the techniques.

II. RELATED WORK

Grid computing and cloud computing are so similar that grid security technique can be applied to cloud computing. Dai et al. made great contribution to Grid security [9-12].

Public Key Infrastructure (PKI) is presently deployed in most grid implementations as it is perceived as a stable and mature technology which is widely supported and can be easily integrated with different applications on various platforms. In the Grid Security Infrastructure (GSI) [13] for the Globus Toolkit (GT) [14], the leading toolkit used in developing grid applications, proxy certificates [15] have been designed and deployed in addition to standard X.509 public key certificate [16], to compensate some of the shortcomings in the conventional PKI setting and to provide additional properties that align with the requirements for secure communications among grid entities within a dynamically changing environment. The motivations for the proxy certificates which carry short-term public keys are twofold: (i) to limit exposure of long-term credentials, and (ii) to enable single sign-on (or unattended authentication) and delegation services. It is not clear, however, if the extensive use of certificates in the hierarchical PKI setting within a dynamic grid environment offers the best possible solution for public key management.

Identity-Based Cryptography (IBC) is in a very quick development [6, 7]. Identity-Based Encryption (IBE) provides a public key encryption mechanism where a public key is an arbitrary string such as an email address or a telephone number [17, 18]. The corresponding private key can only be generated by a Private Key Generator (PKG) who has knowledge of a master secret. Using this construct, anyone can encrypt messages or verify signatures without prior key distribution beyond the dissemination of public parameters and the public key "strings." This is useful where the deployment of a traditional certificate authority-based PKI is inconvenient or infeasible, as IBE-based systems do not require certificate management, eliminating the need for certificate lookups and complex certificate revocation schemes. A central operational consideration of Identity-Based Cryptography is that private keys must be obtained from the PKG. How one securely and efficiently obtains this private key is essential to the security of the supported system. For example, how the PKG decides who should be given the private key associated with an email address is crucial to maintaining the integrity of the system. Another consideration is cost: key generation can be computationally expensive. To ease the computation burdens of PKG operation, hierarchical IBE (HIBE) [19, 20] can be used to reduce the overload of a root PKG by replicating private key generation to slave PKGs. Recently, Waters presented a dual system encryption, which opened up a new way to prove security of IBE and related encryption systems[6]. Boneh provided a general framework for constructing identity-based and broadcast encryption systems, which solves the application problem of identity-based encrypted e-mail[7]. There are other applications, e.g.[21][22].

The idea of applying IBC to grid security was initially explored by Lim [23]. However, the supposedly dynamic use of identity-based keys has been hindered by some traditional limitations of IBC such as key escrow and the need to distribute private keys through secure channels. In addition, the proposals in [23] do not address some of the essential security requirements desired in the GT such as using proxy credentials for single sign-on and delegation.

Mao et al. proposed an identity-based non-interactive authentication framework for grid [24]. The framework improves the user side performance for the current GSI authentication scheme in a considerable degree. The performance improvement is in both computation and communication. The improvement in communication due to being able to batch authentication sessions via a resource broker is significant. However, the authentication framework did not study hierarchy so that the unique Private Key Generator (PKG) becomes the bottleneck of the framework.

Lim and Robshaw proposed a hybrid approach combining identity-based techniques at the user level and traditional PKI to support key management above the user level [25]. In this hybrid setting, each user publishes a fixed parameter set through a standard X.509 certificate; this parameter set then allows users to act as their own Trusted Authorities for the purposes of delegation and single sign-on. This framework solves the two issues of key escrow and distribution of private keys in IBC, but has the limitation that the original dynamic and lightweight qualities that IBC offers are partially lost, because users now need to authenticate and verify other parties' parameter sets before they can be used.

Chen et al. [26] revisited the GSI in the GT version 2 (GT2) and presented some improvements to the security architecture. Their work is related to [25] in which each user has a static long-term credential which can be used by other parties to derive dynamic public keys on-the-fly. Chen et al. modified the security protocols in [13] and the improved protocols seem to offer better performance. In addition, they also proposed an interesting application of aggregate signature to save computational costs in verifying chained signatures. As with [25], however, each user is required to get hold of the intended communicating party's authentic certificate before a dynamic public key can be computed and used.

To the best of our knowledge, there are only a few attempts to apply IBC to cloud computing. Yan et al. [27] provided federated identity management in the cloud such that each user and each server will have its own unique identity, and the identity is allocated by the system hierarchically. With this unique identity and Hierarchical Identity-Based Cryptography (HIBC), the key distribution and mutual authentication can be greatly simplified. Schridd et al. [28] proposed a novel identity-based cryptographic system to avoid the complexity and management problems of certificate-based security infrastructures. However, those works did not study identity-based encryption and signature, and did not make performance analysis and simulation.

In this paper, we first present the Hierarchical Architecture for Cloud Computing (HACC). Then, Identity-Based Encryption (IBE) and Identity-Based Signature (IBS) for HACC are proposed. Finally, an Authentication Protocol for Cloud Computing (APCC) is constructed based on HACC, IBE and IBS. APCC aligns well with the demands of cloud computing. Through simulation experiments, it is shown that APCC is more lightweight and efficient than SAP. The lightweight achieved on the user side is especially significant. The merit of our model in great scalability matches well with the needs of massive-scale cloud.

III. PRELIMINARIES

In this section we briefly review the bilinear pairing. Let G_1 be a cyclic additive group of prime order q, and G_2 be a cyclic multiplicative group of the same order q. A bilinear pairing is a map $\hat{e}: G_1 \times G_1 \rightarrow G_2$ with the following properties:

–Bilinearity: $\forall P, Q \in G_1, \forall a, b \in \mathbb{Z}_q$, we have

$$\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab} \quad (1)$$

–Non-degeneracy: There exist $P, Q \in G_1$ such that

$$\hat{e}(P, Q) = 1 \quad (2)$$

–Computability: There exists an efficient algorithm to compute $\hat{e}(P, Q)$ for $\forall P, Q \in G_1$.

IV. HIERARCHICAL ARCHITECTURE FOR CLOUD COMPUTING

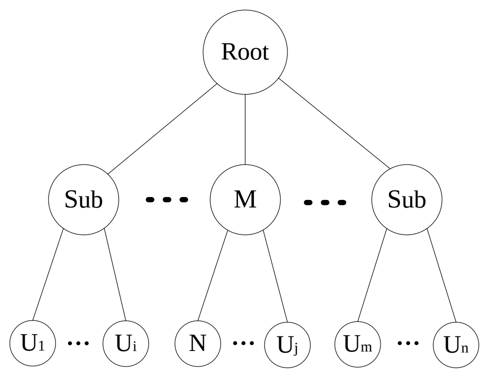

Fig. 1. Hierarchical architecture for cloud computing

As shown in Fig.1, the Hierarchical Architecture for Cloud Computing (HACC) is composed of three levels. The top level (level-0) is root PKG. The level-1 are sub-PKGs. Each node in level-1 corresponds to a data-center (such as a Cloud Storage Service Provider) in the cloud computing. The bottom level (level-2) are users in the cloud computing. In HACC, each node has a unique name. The name should be the node's registered Distinguished Name (DN) when it joins the cloud storage service. For example, in Fig.1, DN of the root node is DN_0, DN of node M is DN_M and DN of node N is DN_N. We define the identity of a node is the DN string from the root to the node. For example, the identity of node N in Fig 1 is

$$ID_N = DN_0 \,||\, DN_M \,||\, DN_N \tag{3}$$

Where"$||$" denotes string concatenation. We further define

$$ID_N\,|_0 = DN_0 \tag{4}$$

$$ID_N\,|_1 = DN_0 \,||\, DN_M \tag{5}$$

$$ID_N\,|_2 = DN_0 \,||\, DN_M \,||\, DN_N \tag{6}$$

The rule is applicable to all nodes in the hierarchical architecture.

The deployment of HACC needs three modules: Root PKG setup, Lower-level setup and User-level setup.

Root PKG setup: Root PKG acts as follows:

1. It generates the groups G_1, G_2 of some prime order q and an admissible pairing

$$\hat{e}: G_1 \times G_1 \rightarrow G_2 \tag{7}$$

2. It chooses cryptography hash functions

$$H_1: \{0,1\}^* \rightarrow G_1 \tag{8}$$

$$H_2: G_2 \rightarrow \{0,1\}^n \tag{9}$$

for some n;

3. It selects a random $S_0 \in \mathbb{Z}_q^*$ and set

$$P_0 = H_1(DN_0) \tag{10}$$

$$Q_0 = S_0 P_0 \tag{11}$$

The root PKG's master key is S_0 and the system parameters are $< G_1, G_2, \hat{e}, H_1, H_2, P_0, Q_0 >$.

Lower-level setup: Assume there are X nodes in the level-1. For each node, the root PKG acts as follows (let M be an arbitrary node in the X nodes):

1. Compute the public key of node M:

$$P_M = H_1(ID_M) \tag{12}$$

Where

$$ID_M = DN_0 \,||\, DN_M \tag{13}$$

2. Set the secret key of node M:

$$S_M = S_0 P_M \tag{14}$$

3. It selects the secret element $\rho_M \in \mathbb{Z}_q^*$ for node M. ρ_M is only known by node M and the root PKG;
4. Define the Q-value:

$$Q_{ID_M|1} = \rho_M P_0 \tag{15}$$

After the above four steps are finished, all nodes in the level-1 obtain their secret keys and secret elements, and securely keep them. The public key and Q-value are publicized.

User-level setup: Assume there are Y child nodes for node M. For each node, the node M acts as follows (let N be an arbitrary node in the Y child nodes):

1. Compute the public key of node N:

$$P_N = H_1(ID_N) \tag{16}$$

Where

$$ID_N = DN_0 \,||\, DN_M \,||\, DN_N \tag{17}$$

2. Set the secret key of node N:

$$S_N = S_M + \rho_M P_N \tag{18}$$

3. Pick the secret point $\rho_N \in \mathbb{Z}_q^*$ for node N. ρ_N is only known by node N and node M;
4. Define the Q-value:

$$Q_{ID_N|2} = \rho_N P_0 \tag{19}$$

After the above four steps are finished, all nodes in the level-2 get and securely keep their secret keys and the secret points. The public key and Q-value are publicized.

V. Identity-Based Encryption and Signature for HACC

In the cloud computing, communications among the users are frequent. To achieve the secure communication, it is important to propose an encryption and signature schemes. Therefore, we propose an Identity-Based Encryption (IBE) and Identity-Based Signature (IBS) schemes for HACC in the following.

A. Identity-Based Encryption

IBE is based on the above Root PKG setup, Lower-level setup and User-level setup algorithms. It is composed of encryption and decryption.

Encryption: Assume E_1 and E_2 are two users in the cloud computing. The identity of E_2 is

$$ID_{E_2} = DN_0 \,||\, DN_1 \,||\, DN_2 \tag{20}$$

To encrypt message m with ID_{E_2}, E_1 acts as follows:

1. Computes

$$P_1 = H_1(DN_0 \,||\, DN_1) \tag{21}$$

$$P_2 = H_1(DN_0 \,||\, DN_1 \,||\, DN_2) \tag{22}$$

2. Chooses a random $r \in \mathbb{Z}_q^*$;
3. Outputs the ciphertext

$$C = < rP_0, rP_2, H_2(g^r) \oplus m > \tag{23}$$

Where

$$g = \hat{e}(Q_0, P_1) \tag{24}$$

can be pre-computed.

Decryption: After receiving the ciphertext $C = < U_0, U_1, V >$, E_2 can decrypt C using its secret key by acting as follows:

$$S_{E_2} = S_0 P_1 + \rho_1 P_2 \tag{25}$$

Where ρ_1 is the secret point of node $DN_0 \,||\, DN_1$:

1.Computes

$$d = \frac{\hat{e}(U_0, S_{E_2})}{\hat{e}(Q_{ID_{E_2}|1}, U_1)} \tag{26}$$

Where

$$Q_{ID_{E_2}|1} = \rho_1 P_0 \tag{27}$$

2. Outputs the message

$$m = H_2(d) \oplus V \tag{28}$$

B. Identity-Based Signature

IBS is also based on the above Root PKG setup, Lower-level setup and User-level setup algorithms. It incorporates two algorithms: signature and verification.

Signature: Assume E_1 and E_2 are two users in the cloud computing. The identity of E_2 is

$$ID_{E_2} = DN_0 \,||\, DN_1 \,||\, DN_2 \tag{29}$$

To sign message m, E_2 acts as follows:

1. Computes

$$P_1 = H_1(DN_0 \,||\, DN_1) \tag{30}$$

$$P_2 = H_1(DN_0 \,||\, DN_1 \,||\, DN_2) \tag{31}$$

$$P_m = H_1(DN_0 \,||\, DN_1 \,||\, DN_2 \,||\, m) \tag{32}$$

2. Computes

$$\delta = S_{E_2} + \rho_2 P_m \tag{33}$$

Where ρ_2 is the secret point of E_2;

3. Outputs the signature $< \delta, P_m, Q_{ID_{E_2}|1}, Q_{ID_{E_2}|2} >$.

Verification: Other users can verify the signature by acting as follows: Confirm

$$\hat{e}(P_0, \delta) = \hat{e}(Q_0, P_1)\ \hat{e}(Q_{ID_{E_2}|1}, P_2)\ \hat{e}(Q_{ID_{E_2}|2}, P_m) \tag{34}$$

Where

$$Q_{ID_{E_2}|1} = \rho_1 P_0 \tag{35}$$

$$Q_{ID_{E_2}|2} = \rho_2 P_2 \tag{36}$$

If the equation (34) is true, the signature is validated.

VI. An Authentication Protocol for Cloud Computing

In this section, based on the former IBE and IBS schemes, a secure Authentication Protocol for Cloud Computing (APCC) is proposed. APCC is analogous to the TLS protocol which uses the RSA key exchange algorithm as specified in [29].

(1) $C \rightarrow S$: **ClientHello (** $n_C, SID, specification_C$)
ClientHelloDone
(2) $S \rightarrow C$: **ServerHello (** $n_S, SID, specification_S$)
ServerHelloDone
(3) $C \rightarrow S$: **ClientKeyExchange** ($E_{P_S}[S_{CS}]$)
IdentityVerify ($Sig_{S_C}[M]$)
ClientFinished
(4) $S \rightarrow C$: **ServerFinished(** $Ver_{ID_C}(Sig_{S_C}[M])$, $D_{S_S}(E_{P_S}[S_{CS}])$ **)**

Fig. 2. Authentication for Cloud Computing

Where

C : Client

S : Server

n_C, n_S : The fresh random number

SID : The session identifier

$specification_C$: The cipher specification of C

$specification_S$: The cipher specification of S

S_{CS} : A pre-master secret used to generate the shared key

$E_{P_S}[S_{CS}]$: Encrypt S_{CS} with the public key P_S of S using the encryption algorithm of IBE

M : All handshake messages since the ClientHello message

$Sig_{S_C}[M]$: Sign M with the private key S_C of C using the signature algorithm of IBS

$Ver_{ID_C}(Sig_{S_C}[M])$: Verify the signature $Sig_{S_C}[M]$ with the help of ID_C using the verification algorithm of IBS

$D_{S_S}(E_{P_S}[S_{CS}])$: Decrypt the $E_{P_S}[S_{CS}]$ with the private key S_S using the decryption algorithm of IBE.

As shown in Fig.2, in step (1) , the client C sends the server S a ClientHello message. The message contains a fresh random number n_C , session identifier SID and $specification_C$. $Specification_C$ extends from TLS to handle the IBE and IBS schemes. For example, $Specification_C$ could be the form $TLS\_IBE\_IBS\_WITH\_SHA\_AES$. IBE and IBS are used as secure transporting and authentication. SHA is the hash function. AES is the symmetric encryption algorithm. The ClientHelloDone message means the step (1) finishes.

In step (2), the server S responds with a ServerHello message which contains a new fresh random number n_S , the session identifier SID and the cipher specification $specification_S$. The $specification_S$ is the suie of ciphers supported by S . The ServerHelloDone message means the step (2) is over.

In step (3), C chooses a pre-master secret S_{CS} and encrypts it with the public key P_S of S using the encryption algorithm of IBE. The ciphertext is transmitted to S as ClientKeyExchange message. Then C generates a signature $Sig_{S_C}[M]$ as the IdentityVerify message and forwards it to S . Finally, The ClientFinished message means the step (3) finishes.

In step (4), S firstly verifies the signature $Sig_{S_C}[M]$ with the help of ID_C . C can pass the verification only if it is the valid owner of ID_C . This completes the authentication of C by S .Then S decrypts the $E_{P_S}[S_{CS}]$ with its private key S_S . Because of the fresh S_{CS} , the correct decryption indicates S is the valid owner of ID_S . This step authenticates the validity of S . The ServerFinished message means the step (4) finishes.

Eventually, a shared secret key between C and S is calculated by $K_{CS} = PRF(S_{CS}, n_C, n_S)$, where PRF is a Pseudo-Random Function.

VII. Performance Analysis

In this section, performance comparisons between SAP and APCC are discussed.

A. Computation Cost

The comparison of computation cost between the two different protocols is shown in table Ⅰ. Note that only dominant computation is considered, i.e. encryption, decryption and authentication.

Table Ⅰ
Comparison of Computation Cost

	SAP	APCC
Client	1 ENC_R , 1 SIG_R and Authenticating server	1 ENC_I and 1 SIG_I
Server	1 DEC_R , 1 SIG_R and Authenticating client	1 DEC_I and 1 VER_I

Where

ENC_R = RSA encryption,

DEC_R = RSA decryption,

ENC_I = IBE encryption,

DEC_I = IBE decryption,

SIG_R = RSA signature,

SIG_I = IBS signature,

VER_I = IBS signature verification,

Authenticating server=Including building certification path of server and verifying signatures,

Authenticating client= Including building certification path of client and verifying signatures.

The paper [8] shows that in the SAP, the computation cost of client is one RSA encryption, one RSA signature and Authenticating server. The computation cost of server is one RSA decryption, one RSA signature and Authenticating client. However, in the APCC, the computation cost of client is one IBE encryption and one IBS signature. The computation cost of server is one IBE decryption and one IBS signature verification.

B. *Communication Cost*

The comparison of communication cost between the two different protocols is shown in table Ⅱ. Note that only dominant communication is considered, i.e. certificates, signed or encrypted messages, which may have the greatest consumptions of the network bandwidth.

Table Ⅱ
Comparison of Communication Cost

SAP		APCC	
Certificate	RSA Signature	IBS Signature	IBE Ciphertext
2	2	1	1

Reference [8] shows that the communication cost of SAP is two public key certificates and two RSA signatures. However, in the APCC, the communication cost is only one IBS signature and one IBE ciphertext.

VIII. SIMULATION AND EXPERIMENT RESULTS

A. *Simulation Platform and Reference*

The platform for simulation experiments is GridSim which is based on Java[30]. Special users and resources can be generated by reconfiguring these interfaces. This aligns well with various users and resources of cloud computing. Furthermore, GridSim is based on SimJava which is a discrete event simulation tool based on Java, and simulates various entities by multiple threads. This aligns well with the randomness of entity action in cloud computing. Therefore, it is feasible to simulate our proposed authentication protocol of cloud computing by GridSim.

The simulation environment is composed of four desktop computers with P4 3.0 GHz CPU, and 4G memory. Certification chain is important for SAP. The shorter it is, the better the performance is. The shortest certification chain includes all the 4 certifications: CA_1, client and CA_2, server. There is a cross authentication for CA_1 and CA_2. It is in this scene that SAP and APCC are compared. Based on openssl0.9.7, SAP is implemented. The pairing algorithm is adapted from [31]. To precisely simulate the network delay, there is 20~40ms waiting time before a message is sent.

B. *Experiment Results and Analysis*

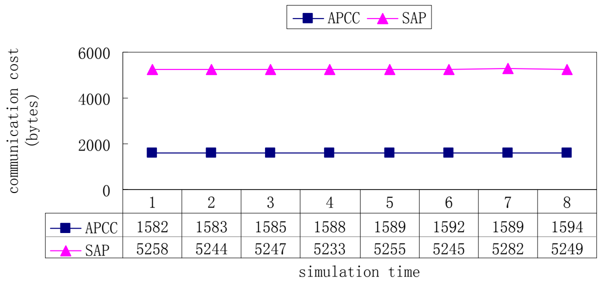

Fig. 3. Comparison of communication cost

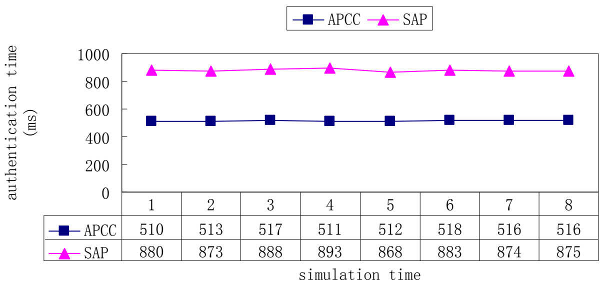

Fig. 4. Comparison of authentication time

As shown in Fig.3, the communication cost of APCC is approximately 1588 bytes while that of SAP is 5252 bytes. That is to say, the communication cost of APCC is 30% of that of SAP. Fig.4 shows the authentication time of APCC is approximately 514 ms while that of SAP is 879 ms. That is, the authentication time of APCC is 58% of that of SAP. The simulation results confirm that the communication cost of APCC is lower and the authentication time is shorter.

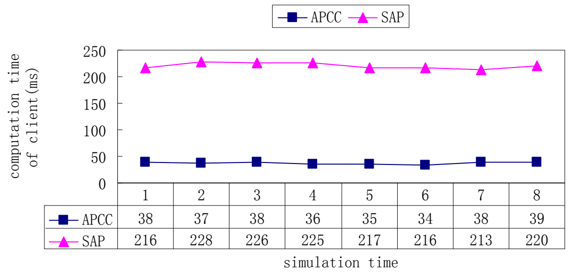

Fig. 5. Comparison of computation time of client

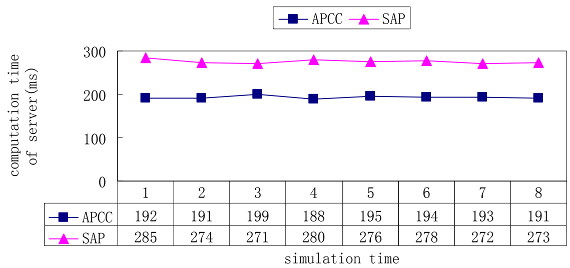

Fig. 6. Comparison of computation time of server

Fig.5 illustrates the computation time of client for APCC is approximately 37 ms while that for SAP is 220 ms. That is to say, the computation time of client for APCC is 17% of that for SAP. Fig.6 illustrates the computation time of server for APCC is approximately 193 ms while that for SAP is 276 ms. Therefore, the computation time of server for APCC is 70% of that for SAP. The simulation results confirm that both client and server of APCC are more lightweight than those of SAP.

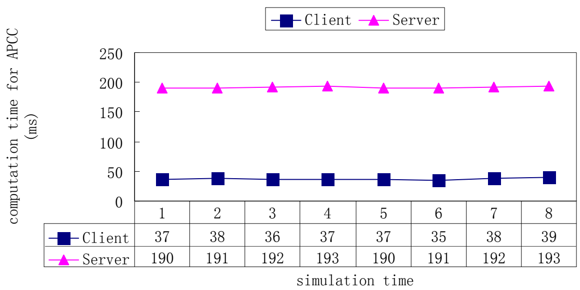

Fig. 7. Comparison of computation time for APCC

As shown in Fig.7, In APCC, the computation time of client is approximately 37 ms while that of server is 192 ms. That is to say, the computation time of client is 19% of that of server in APCC. This aligns well with the idea of cloud computing which allows the user with a platform of limited performance to outsource its computational tasks to some more powerful servers. As a result, the more lightweight user side can connect more servers and contribute to a larger scalability.

IX. Conclusion

Security is significant in cloud computing. In this paper, first, we present a novel Hierarchical Architecture for Cloud Computing (HACC). Then, Identity-Based Encryption (IBE) and Identity-Based Signature (IBS) for cloud computing are proposed. Finally, an Authentication Protocol for Cloud Computing (APCC) is constructed based on HACC, IBE and IBS. Being certificate-free, APCC aligns well with the demands of cloud computing. Through simulation experiments, it is shown that the authentication protocol is more lightweight and efficient than SSL Authentication Protocol (SAP). The lightweight achieved on the user side is especially significant. The merit of our model in great scalability matches well with the needs of massive-scale cloud.

References

[1] H. Erdogmus, "Cloud Computing: Does Nirvana Hide behind the Nebula? " *IEEE Software*, vol. 26, no.2, pp. 4-6 ,2009.

[2] Y. S. Dai, Y. P. Xiang, G. W. Zhang., "Self-Healing and Hybrid Diagnosis in Cloud Computing, " *Lecture Notes of Computer Science (LNCS)*, vol. 5931, pp. 45-56,2009.

[3] Amazon Elastic Compute Cloud [URL].http://aws.amazon.com/ec2, access on Oct. 2009.

[4] IBM Blue Cloud project [URL]. http://www-03.ibm.com/press/us/en/pressrelease/22613.wss/, access on October 2009.

[5] J. Abawajy, "Determining Service Trustworthiness in InterCloud Computing Environments," *10th International Symposium on Pervasive Systems, Algorithms, and Networks (I-SPAN 2009)*, pp.784-788, 2009.

[6] B. Waters, "Dual key encryption: Realizing Fully Secure IBE and HIBE Under Simple Assumption," *In Proc. of CRYPTO'09, Lecture Notes of Computer Science (LNCS)*, vol.5677, pp.619-636, 2009.

[7] D. Boneh, "Generalized Identity Based and Broadcast Encryption Schemes ,"*In ASIACRYPT'08, Lecture Notes of Computer Science (LNCS)*, vol.5350, pp. 455-470 ,2008.

[8] A. O. Freier , P. Karlton, and P. C. Kocher, "The SSL Protocol, Version 3.0," *IETF Internet-Draft* , 1996, http://tools.ietf.org/id/draft-ietf-tls-ssl-version3-00.txt.

[9] Y. S. Dai, Y. Pan, X. K. Zou, "A Hierarchical Modelling and Analysis for Grid Service Reliability,"*IEEE Transactions on Computers*, vol. 56, no. 5, pp. 681-691 ,2007.

[10] Y. S. Dai, G. Levitin, K. S. Trivedi, "Performance and Reliability of Tree-Structured Grid Services Considering Data Dependence and Failure Correlation," *IEEE Transactions on Computers*, vol. 56, no. 7, pp. 925-936 ,2007.

[11] Y. S. Dai, G.Levitin, "Reliability and Performance of Tree-structured Grid Services,"*IEEE Transactions on Reliability*, vol. 55, no.2, pp. 337-349 ,2006.

[12] Y. S. Dai, M. Xie,, X. L. Wang, "Heuristic Algorithm for Reliability Modeling and Analysis of Grid Systems,"*IEEE Transactions on Systems, Man, and Cybernetics*, Part A., vol.37, no. 2, pp. 189-200, 2007.

[13] I. Foster, and C. Kesslman, G. Tsudik, "A Security Architecture for Computational Grids,"*ACM Conference on Computers and Security*, pp. 83-90, 1998.

[14] I. Foster and C. Kesselman, “Globus: A Metacomputing Infrastructure Toolkit,” *International Journal of Supercomputing Applications*, vol.11,no2, pp. 115-128, 1997.

[15] S. Tuecke, V. Welch, D. Engert, L. Pearman, and M. Thompson, “Internet X.509 Public Key Infrastructure Proxy Certificate Profile,”*The Internet Engineering Task Force (IETF)*, RFC 3820, June 2004.

[16] R. Housley, W. Polk, W. Ford, and D. Solo, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,” *The Internet Engineering Task Force (IETF)*, RFC 3280, April 2002.

[17] A. Shamir, “Identity-Based Cryptosystems and Signature Schemes,” *In :Advances in Cryptology - Proceedings of CRYPTO'84, Lecture Notes of Computer Science (LNCS)*, vol.196, pp.47-53, 1985.

[18] D. Boneh and M. Franklin, “Identity Based Encryption From the Weil Pairing,” *In: Advances in Cryptology-Crypto 2001, Lecture Notes of Computer Science (LNCS)*, vol.2139, pp.213-229,2001.

[19] C. Gentry and A. Silverberg, “Hierarchical ID-Based Cryptography,” *In: ASIACRYPT 2002,Lecture Notes of Computer Science (LNCS)*, vol. 2501, pp. 548-566, 2002.

[20] C. Gentry and S. Halevi, “Hierarchical Identity Based Encryption with Polynomially Many Levels,” *In Theory of Cryptography, Lecture Notes of Computer Science (LNCS)*, vol. 5444 , pp.437-456, 2009.

[21] J. Y. Sun, C. Z, Y. C. Zhang, and Y. G. Fang, “An Identity-Based Security System for User Privacy in Vehicular Ad Hoc Networks,”*IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no.9, pp. 1227-1239, 2010.

[22] J. Y. Sun, C. Z, Y. C. Zhang, and Y. G. Fang, “Identity-Based Anonymous Remote Authentication for Value-Added Services in Mobile Networks,”I*EEE Transactions on Vehicular Technology*, vol. 58, no.7, pp. 3508-3517, 2009.

[23] H. W. Lim, and M. Robshaw, “On Identity- Based.Cryptography and Grid Computing,” *In: ICCS 2004, Lecture Notes of Computer Science (LNCS)*, vol.3036, pp. 474-477 ,2004.

[24] W. B. Mao, “An Identity-Based Non-interactive Authentication Framework for Computational Grids,” *http://www.hpl.hp.com/ techreports/2004/HPL-2004-96.pdf* , 2004.

[25] H. W. Lim, and M. Robshaw, “A Dynamic Key Infrastructure for GRID,” *Proceedings of the European Grid Conference (EGC 2005), Lecture Notes of Computer Science (LNCS)*, vol.3470, pp. 255-264, 2005.

[26] L. Chen, H. W. Lim, , and W. B. Mao, “User-friendly Grid Security Architecture and Protocols,” *Proceedings of the 13th International Workshop on Security Protocols* , pp. 505-514,2005.

[27] L. Yan, C. M. Rong, and G. S. Zhao, “Strengthen Cloud Computing Security with Federal Identity Management Using Hierarchical Identity-Based Cryptography,” *In The First International Conference on Cloud Computing*, pp. 167–177, 2009.

[28] C. Schridde, T. Dornemann, E. Juhnke, B. Freisleben, M. Smith, “An Identity-Based Security Infrastructure for Cloud Environments,” *2010 IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, pp. 644 – 649, 2010.

[29] T. Dierks and E. Rescorla, “The Transport Layer Security(TLS) Protocol Version 1.2,” *IETF RFC 5246*,2008, http://www.ietf.org/rfc/rfc5246.txt.

[30] R. Buyya, M. Murshed, “GridSim: A Toolkit for The Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing,”*Journal of concurrency and computation practice and experience*, vol.14 , pp. 1175-1220 ,2002.

[31] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott, “Efficient Algorithms for Pairing-based Cryptosystems,” *Advances in Cryptology–Proceedings of CRYPTO 2002, Lecture Notes of Computer Science (LNCS)*, vol.2442, pp. 354-368 ,2002.