An Iterative Hierarchical Key Exchange Scheme for Secure Scheduling of Big Data Applications in Cloud Computing

Chang Liu†, Xuyun Zhang†, Chengfei Liu*, Yun Yang*, Rajiv Ranjan‡, Dimitrios Georgakopoulos‡ and Jinjun Chen†

†Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia
*Faculty of Information and Communication Technologies, Swinburne University of Technology, Australia
‡Information Engineering Laboratory, CSIRO ICT Centre, Australia
Email: {changliu.it, xyzhanggz, jinjun.chen, rranjans}@gmail.com; {cliu, yyang}@swin.edu.au; dimitrios.georgakopoulos@csiro.au

***Abstract* — As the new-generation distributed computing platform, cloud computing environments offer high efficiency and low cost for data-intensive computation in big data applications. Cloud resources and services are available in pay-as-you-go mode, which brings extraordinary flexibility and cost-effectiveness as well as zero investment in their own computing infrastructure. However, these advantages come at a price - people no longer have direct control over their own data. Based on this view, data security becomes a major concern in the adoption of cloud computing. Authenticated Key Exchange (AKE) is essential to a security system that is based on high efficiency symmetric-key encryption. With virtualization technology being applied, existing key exchange schemes such as Internet Key Exchange (IKE) becomes time-consuming when directly deployed into cloud computing environment. In this paper we propose a novel hierarchical key exchange scheme, namely Cloud Background Hierarchical Key Exchange (CBHKE). Based on our previous work, CBHKE aims at providing secure and efficient scheduling for cloud computing environment. In our new scheme, we design a two-phase layer-by-layer iterative key exchange strategy to achieve more efficient AKE without sacrificing the level of data security. Both theoretical analysis and experimental results demonstrate that when deployed in cloud computing environment, efficiency of the proposed scheme is dramatically superior to its predecessors CCBKE and IKE schemes.**

Keywords - cloud computing; big data; key exchange; efficient security-aware scheduling; virtualisation security

I. INTRODUCTION

As the new generation computing paradigm, cloud computing has been attracting extensive interest from IT industry and academia in recent years. One of its core concepts is 'X as a Service' (XaaS), including Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS), which means that both individual and enterprise users can use IT services in pay-as-you-go fashion. Compared to traditional distributed systems, this new concept in cloud computing brings outstanding advantages. First, a considerable amount of investment is saved, for there is no need for users to purchase and maintain their own IT facilities. Second, it brings exceptional elasticity, scalability and efficiency for task executions, especially in big data applications [6, 9]. Therefore, utilising cloud to run data-intensive applications, such as applications in scientific research or social networks, becomes a cost-effective choice. For example, the Swiss physics research lab CERN [14] utilised cloud computing to help execute the computation tasks for their particle accelerator which generates 22PB data every year. Until now, there have been several cloud computing projects that provide public cloud computing services for scientific users, for example Nimbus [19]. By utilising cloud computing services, the numerous capital investments in building and maintaining a supercomputing or grid computing environment for big data applications can be effectively reduced.

Despite these advantages, data security concern is still one major obstacle in migrating big data applications to cloud computing environments. For example, data in online social systems are highly privacy-sensitive; data used for scientific research often contain private information and/or unpublished results, which make them valuable intellectual properties. When submitting tasks to the cloud, people also surrendered control over their valuable data at the same time. The convenience of cloud computing is always accompanied by the risk of data exposure, which is not the case in traditional distributed computing systems where data are stored locally.

In order to ensure data security against a malicious third party, encryption is a common choice. Fully-homomorphic encryption can save all the trouble of key management, in that it renders the encrypted dataset to be still decrypt-able with the key used for encryption even after any kind of operation has been performed on it. Unfortunately, to the best of our knowledge, developing a fully-homomorphic encryption algorithm with practically feasible time-complexity still remains an open research problem [12]. Asymmetric-key encryption does not require key exchange, but it is way too

time-consuming when employed over large datasets. In order to utilise the much faster symmetric-key encryption, Authenticated Key Exchange (AKE) scheme is a mandatory requirement -- not only for negotiating session keys used in encryption, but also for mutual authentication to deny fake-ID attacks / Man-in-the-Middle (MITM) attacks. Reducing time consumption of AKE schemes has hardly been an attractive research problem in the past. However, in cloud computing, at least thousands of virtualised server instances, i.e., virtual machines (VMs), are participating in the execution of almost every task, where the most widely-used Diffie-Hellman (DH) based AKE schemes can become very time consuming. When deploying IKE along with the execution of large-scale tasks in cloud environment, it can even take up to 76% of the total time consumed by the security system [22].

In this paper, we propose a hierarchical AKE scheme – Cloud Background Hierarchical Key Exchange (CBHKE), in the aim of secure and efficient scheduling in cloud computing for large-scale computing tasks such as scientific applications. Our work is based on our previously proposed scheme named Cloud Computing Background Key Exchange (CCBKE) and its predecessor, the standardised Internet Key Exchange (IKE) scheme. All these schemes belong to the DH-AKE family.

The rest of this paper is organised as follows. Section II discusses related work. Section III provides a motivating example as well as detailed analysis to our research problem. Section IV gives the description of our CBHKE scheme in detail based on a series of notations, followed by the analysis of security and efficiency. Section V evaluates the efficiency of our scheme through experimental results. Section VI concludes our work and discusses future work.

II. Related Work

Nowadays, because of its outstanding cost-effectiveness [10], cloud computing is being widely utilised for large-scale computation tasks in big data processing [6, 9] such as scientific research applications [14, 19]. Great efforts have been placed on cost-efficient scheduling for distributed computing systems including cloud computing [11, 25, 27, 28]. The work of Garg et.al [11] and Lee et.al [27] focused on efficient scheduling with low energy consumption in cloud computing and distributed systems; Yuan et.al [28] investigated the trade-off between the cost of storage and computing in cloud scheduling. However, none of their work has taken into account the additional cost in enhancing data security which is another important metric of QoS. Tang et.al [25] proposed a cost-effective security-aware scheduling algorithm for distributed systems. Although their approach achieved high efficiency by grading data security into several levels, their scheme in fact compromised security for higher efficiency, and it still suffered from the inherent cost-inefficiency of existing KE schemes. The security-aware scheduling scheme for distributed computing systems proposed by Xie et.al [26] suffered from similar problems.

Key exchange (KE) over untrusted communication environment has always been an active research topic in public-key cryptography. In [21, 29], extended security standards were formalised and researched for key exchange schemes for the basic 2-party scenario. For efficiency, some of the most recent research on authenticated key exchange schemes focused on password-based key exchange [13, 15], which allows two parties to share a session key through exchanging a low-entropy password. Many existing key exchange schemes were trying to optimise the multi-party-same-key scenario as we do. For example, some of them focused on the scenario with users join and leave dynamically, such as [16, 30]. Kurosawa [20] and then Bellare et.al [7] studied the problem of asymmetric-key encryption in multi-user-different-data scenario with randomness reuse strategy. This problem is essentially the same as KE in the background of cloud. Due to the low efficiency of asymmetric-key encryption over large datasets, their schemes cannot be directly applied into cloud computing environments. However, the ideas from these schemes inspired our work directly.

Internet Key Exchange (IKE) [17], as the standardised key exchange solution, has been widely adopted along with IPSec to ensure data confidentiality, integrity and authenticity while data is transferred via Internet. Its security has been formally proven [8]. In the past, we developed a key exchange scheme named CCBKE for security-aware scheduling in cloud computing [22]. When deployed in cloud, CCBKE invokes significantly lower time consumption compared to IKE, but it still takes a considerable amount of time. We observed that in CCBKE, the central cloud controller needs to generate keying materials and compute all session keys (to be used to communicate with ALL virtualised instances later on) by itself, and the computational power of intermediate control nodes are not effectively utilised. This is the main motivation of our research in this paper.

III. Motivating Example and Problem Analysis

A. Motivating Example and Research Problem

Big data applications such as scientific applications are always data-intensive and time-critical. Take astrophysics research for example, Australian astrophysics researchers operate a gigantic 64-metre Parkes telescope [1] which generates a large amount of data through constant observation. In pulsar searching workflow [28] the raw observation data are generated at a 1GB-per-second ratio. The beam file is generated for later pulsar searching through a local real-time compression operation on the raw data file. During this process, about 1 to 20GB beam file is generated from 4 to 60 minute observation. Scientists usually need to access the results as early as possible, as a late-coming result may cause an enormous waste of resources and loss of scientific discovery. Example in astrophysics is gravitational wave detection [18] which is especially time-critical. Due to its nature of real-time and streaming, delay in returning a result of a task may cause missed detection of an incoming gravitational wave. This will lead to loss of scientific discovery as the next wave would come after years. Moreover, thousands of hours of data-intensive computation was in vain, which is a terrible waste both economically and environmentally. Online web service is another example,

where user requests normally demand servers' response in a few seconds. Hence, efficiency is of extreme importance in cloud scheduling for many big data applications.

Cloud computing's unique characteristics of virtualisation, consolidation and multi-tenancy bring unpredictable challenges to data security. For example, a malicious party can easily be another legitimate user who is using the same cloud and has even more advantage for successful malicious behaviours [24]. As discussed in Section I, Data in scientific research are valuable intellectual property which can either be people's privacy-sensitive information or directly related to scientific discovery. Therefore, we suggest that all user data always stay encrypted in cloud. Decryptions may only be applied right before data are used for task execution.

In big data applications, each interaction between users and cloud servers requires key exchange because of encryption. As a result, a large percentage of time overall is devoted into the security system. As demonstrated in [22], the standardised IKE key exchange scheme can take up to 76% of the total time consumption in the security system (depending on the actual parameters) when the size of user datasets and the number of instances involved are large, which is why we need to improve the efficiency of key exchange schemes.

B. Problem Analysis

In every KE session, a distinct session key is needed for every virtual machine. This is because the risk of additional information being exposed against malicious users needs to be minimised. The existence of virtual machine hijacks [24] further intensifies this risk. For example, if a single session key is utilised for data encryption on 100 virtualised instances, the information on all 100 nodes will all be exposed when only one of the instances is hijacked and the key is revealed. If we use different keys for different instances, the total information leakage will be reduced by 99%. For this reason, the computation cost and time consumption of key exchange operations in cloud are more significant than in other distributed computing systems.

Computations on server instances in key exchange processes can be completed almost instantly. Data communications in KE schemes via networking take almost no time as well because only kilobytes of data need to be transferred between the cloud controller and server instances in order to complete key exchange. Digital signatures are always necessary in key exchange schemes for identity authentication. In key exchange schemes, messages to be signed are usually of a short fixed length (typically 128 bits which is the output size of a HMAC function). In this regard, time consumption in signing and verification of messages is negligible when compared to modular exponentiations over 1024-bit keying materials related to key exchanges. Based on this view, we know that the modular exponentiations in KE operations are acting as the predominant factor in the efficiency for a distributed KE scheme.

For scheduling purposes, a large-scale cloud computing infrastructure often employs a hierarchical control structure. Following the acronyms defined in the early Eucalyptus cloud system [23], a typical cloud computing structure may employ a CLC (cloud controller) as the interface between user and cloud, several CC (cluster controllers) for cluster control, a bunch of NC (node controllers) for virtualisation, and then virtualised instances for actual task execution. These are at least three layers for control, and the number of control layers could increase further with the scaling of the cloud environment (see Figure 1 for an example of a hybrid cloud which is consisted of multiple clouds with multiple control layers). In CCBKE[22], CLC needs to perform all the KE operations for exchanging a distinct key for each instance, while the intermediate layers are required to do nothing other than passing the messages. In this regard, the efficiency of KE will be further improved by re-designing the scheme to distribute the modular exponentiations to other control nodes.

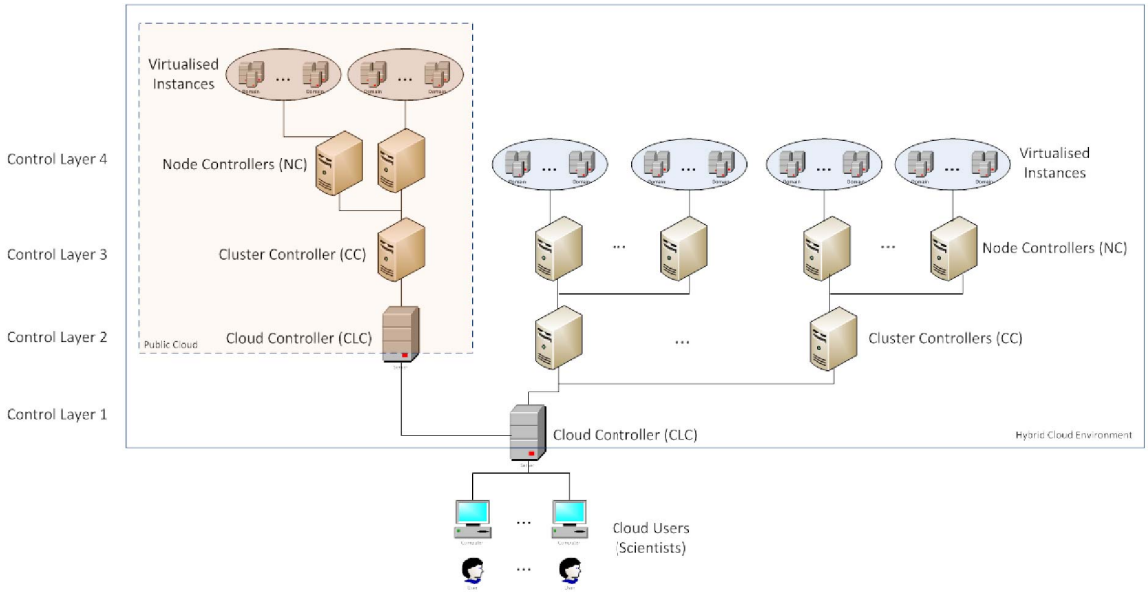

Fig.1 An example of a hybrid cloud structure used for scientific applications

IV. CBHKE – Cloud Background Hierarchical Key Exchange

Before showing how the CBHKE scheme works, we first define the symbolised notations to be used for the formalised description.

A. Notations

$N_{i,j}$:	The j th control node (or *node*, for simplicity) on layer i. $N_{1,1}$ is the cloud controller (CLC)
HDR_i :	Header, contains security parameter indexes
$CertReq$:	Certificate request
$Cert_N$:	Certificate of node N
$nonce$:	One-time nonce for message freshness
SA:	Security associations, used in negotiating cryptographic algorithms
ID_N:	Identity information of node N
Sig_N:	Node N's signature, which can be verified using algorithms negotiated in SA and public key in $Cert$
prf():	Pseudorandom function
$\{M\}_k$:	Encrypt message M with session key k
$DEC_k(M)$:	Decrypt message M with session key k
l:	Total number of control layers
n_i:	Number of nodes on the i th layer
$m_{i,j}$:	Number of sub-nodes for node j on the ith layer
PUB_N:	Node N's public key for KE
$\overline{N}_k$:	N's sub-nodes (children nodes), where $k = 1, \dots, m$
$\widetilde{N}$:	N's parent node
KEY_UP_N:	Temporary key used by node N to encrypt the communications with its parent node
$KEY\_DOWN_{N,k}$:	Temporary key used by node N to encrypt the communications with its kth sub-node

B. System Setup

The system chooses a large prime integer p and select a generator g of group $\mathbb{Z}_p^*$. Normally p is a Sophie Germain prime where $(p-1)/2$ is also prime, so that the group Z_p^* has maximum resilience against square root attack. A certificate authority (CA) is needed in our security system so that communicating parties can identify each other through exchanging verifiable certificates $Cert_N$, although we will skip further details regarding these straightforward CA-involved operations in detailed scheme description.

C. Key Exchange

This is a generalised description for a cloud infrastructure that has l control layers, from CLC to end NC. Layer i has n_i nodes, namely $N_{i,j}$, $i = 1, \dots, l$, $j = 1, \dots, n_i$. CLC is on layer 1, where $n_1 = 1$. Let $m_{i,j}$ be the numbers of sub-nodes for nodes $j = 1, \dots, n_i$ on layer i.

Overview: The scheme can be divided into two phases. The aim of Phase 1 is for the CLC to securely deliver its secret keying material to NCs, while Phase 2 is for the actual key exchange. In Phase 2, CLC deliver to NCs interact with virtualised instances on CLC's behalf and send back the results of KE to the CLC.

Phase 1: This phase is for KE between all control nodes from CLC (layer L_1) to the lth control layer (layer L_l, i.e., NC layer). This exchanged session key will be used for encrypting the real keying material in Phase 2.

All control nodes picks their own private key $x_{i,j}$ and one-time nonce $nonce_{i,j}$. They compute their public key for KE as follows:

$$PUB_{N_{i,j}} = g^{x_{i,j}}$$

Then CLC broadcasts the very first message $MSG\_1_{1,1}$:

$$MSG\_1_{1,1}: REQ, HDR_{1,1}, SA, PUB_{N_{1,1}}, nonce_{1,1}$$

to all nodes in layer 2. REQ is a flag for message identification, indicating the request for keying material.

For the nodes $N_{i,j}$ $(j = 1, \dots, n_i)$ in layer L_i , upon receiving message $MSG\_1_{(i-1),j}$ from their parent-node in L_{i-1} $(i = 2, \dots, l)$, they send messages $MSG\_1_{i,j}$ to their sub-nodes in the next layer L_{i+1}:

$$MSG\_1_{i,j}: REQ, HDR_{i,j}, SA, PUB_{N_{i,j}}, nonce_{i,j}$$

Meanwhile, they respond MSG_2 to their parent node:

$$MSG\_2_{i,j}: REQ, HDR_{i,j}, SA, PUB_{N_{i,j}}, CertReq, nonce'_{i,j}$$

After receiving MSG_1, every node in layers $L_2, \dots, L_l$ will know its parent node $\widetilde{N}$'s public key $PUB_{\widetilde{N}}$, and compute the session key for communicating with its parent:

$$KEY\_UP_{N_{i,j}} = (PUB_{\widetilde{N}})^{x_{i,j}},$$

where $i = 2, \dots, l; j = 1, \dots, n_i; k = 1, \dots, m_{i,j}$

For nodes in layer L_{i-1}, upon receiving MSG_2 from their sub-nodes, they'll know the public keys of their sub-nodes, namely $g^{x_{i+1,j}}$. We denote the public key of node $N_{i,j}$'s sub-nodes $\overline{N}_k$ as $PUB_{\overline{N}_k}, k = 1, \dots, m_{i,j}$. $N_{i,j}$ compute the following session keys for communicating with their sub-nodes:

$$KEY\_DOWN_{N_{i,j},k} = \left(PUB_{\overline{N}_k}\right)^{x_{i,j}},$$

where $i = 1, \dots, l-1; j = 1, \dots, n_i; k = 1, \dots, m_{i,j}$

For authentication, all nodes in L_i $(i = 1, \dots, l-1)$ broadcast MSG_3 to its sub-nodes $\overline{N}_k$:

$$MSG\_3_{i,j}:$$
$$HDR_{i,j}, \left\{ID_{N_{i,j}}, SA, Cert_{N_{i,j}}, CertReq_{\overline{N}_1}, \mathrm{Sig}_{N_{i,j}}\right\}_{KEY\_DOWN_{N_{i,j},1}}$$

$||\ldots$

$||\left\{ID_{N_{i,j}}, SA, Cert_{N_{i,j}}, CertReq_{N_k}, \mathrm{Sig}_{N_{i,j}}\right\}_{KEY\_DOWN_{N_{i,j},k}}$

$(i = 1, \ldots, l-1; j = 1, \ldots, n_i; k = 1, \ldots, m_{i,j})$

where the structure of message for signatures is also an output of prf(), similar to IKE. All nodes on L_i, $(i = 2, \ldots, l)$ will receive this message, and respond with $MSG\_4_{i,j}$ if signature verification is successful:

$MSG\_4_{i,j}$:

$HDR_{i,j}, \left\{ID_{N_{i,j}}, SA, Cert_{N_{i,j}}, \mathrm{Sig}_{N_{i,j}}\right\}_{KEY\_UP_{N_{i,j}}}$

$(i = 2, \ldots, l; j = 1, \ldots, n_i)$

The reason that only the receiver of MSG_3 and MSG_4 can decrypt them is that, for every parent-child node pair N_{PARENT} and N_{CHILD}, we already have:

$$KEY\_DOWN_{N_{\mathrm{PARENT}}} = KEY\_UP_{N_{\mathrm{CHILD}}}$$

which concludes phase 1.

Phase 2: This phase is for the eventual goal of our scheme – KE between CLC and virtualised instances. The outcome of Phase 1 will play a vital role here.

CLC picks its secret value X as its keying material for KE with those virtualised instances. CLC encrypts X with the session key negotiated in phase 1 and broadcast the following message to the next layer:

$MSG\_1_{1,1}$: $\{X||ID_{N_{1,1}}\}_{KEY\_DOWN_{N_{1,1},1}}||\ldots$
$||\{X||ID_{N_{1,1}}\}_{KEY\_DOWN_{N_{1,1},m_{1,1}}}$

Upon receiving message $MSG\_1_{(i-1),j}$ from their parent-nodes in L_{i-1} $(i = 2, \ldots, l)$, the nodes in L_i broadcast similar $MSG\_1_{i,j}$ to their sub-nodes in L_{i+1}:

$MSG\_1_{i,j}$: $\{X||ID_{N_{i,j}}\}_{KEY\_DOWN_{N_{i,j},1}}||\ldots$
$||\{X||ID_{N_{i,j}}\}_{KEY\_DOWN_{N_{i,j},m_{i,j}}}$

because the recipients can obtain X by decrypting the received MSG_1 using its $KEY_{UP_{N_{i,j},k}}$. For security reasons, all nodes in $L_2, \ldots, L_{l-1}$ should destroy X after sending MSG_1 in Phase 2 where they re-encrypt X with KEY_{DOWN} and send to their sub-nodes.

After these operations, once nodes on layer L_l, i.e., NCs, get to know the X value. They now use this secret value to perform a 4-round CCBKE to finish the final KE:

NC-VM: $HDR_{NC}, SA_{NC}, g^X, nonce_c$
VM-NC: $HDR_{VM_i}, SA_{VM_i}, g^{Y_i}, nonce_{S_i}, CertReq$
NC-VM: HDR_{NC},
$\{ID_{NC}, SA_{NC}, Cert_{NC}, CertReq, \mathrm{Sig}_{NC}\}_{g^{XY_1}}$
$||\ldots||\{ID_{NC}, SA_{NC}, Cert_{NC}, CertReq, \mathrm{Sig}_{NC}\}_{g^{XY_\alpha}}$
VM-NC: $HDR_{VM_i}, \left\{ID_{VM_i}, SA_{VM_i}, Cert_{VM_i}, \mathrm{Sig}_{VM_i}\right\}_{g^{XY_i}}$

The final session key for data encryption is g^{XY_i} where $i = 1, \ldots, \alpha$. After this step, not only α but all $m_{i,j}$ virtualised instances will have the desired session key for data encryption/decryption.

Now all virtualised instances have exchanged a key with their control nodes. For each NC, i.e. $N_{l,j}$ $(j = 1, \ldots, n_l)$, they combine and encrypt the final session keys in this format:

$MSG\_2_{l,j}$: $\{g^{XY_1}||\ldots||g^{XY_{m_{l,j}}}\}_{KEY\_UP_{N_{l,j}}}$

and send it to its upper level. Then, nodes in every level from L_i, $i = 2, \ldots, i-1$ compute and send the following message to their parent nodes, after receiving from their sub-nodes:

$MSG\_2_{i,j}$:
$\{DEC_{KEY\_DOWN_{N_{i,j},1}}(MSG\_2_{N_{i+1,1}})||\ldots$
$||DEC_{KEY\_DOWN_{N_{i,j},m_{i,j}}}\left(MSG\_2_{N_{i+1,m_{i,j}}}\right)\}_{KEY\_UP_{N_{i,j}}}$

After this layer-by-layer action, CLC, i.e., $N_{1,1}$, will know the session keys g^{XY_α} that has been negotiated with all virtualised instances, thereby concludes the KE scheme. The task data stored at CLC can now be split, encrypted and distributed to the virtualised instances for execution. After the execution, the server instances may follow an inversed procedure to exchange session keys with CLC and send back the encrypted results.

D. Security Analysis:

The security of our scheme is analysed in Dolev-Yao's threat model with a bit extension. As we are dealing with communication security only, all the data stored on CLC and intermediate control nodes are assumed safe against the adversary in this model. This is a practical assumption because usually the control nodes are not directly exposed to users and the open network environment, which renders them less vulnerable than the processing servers and virtualised instances. The abilities of the adversaries, or attackers, are defined as follows:

Definition 1 *(attackers):* A cloud outside attacker M_o aims to retrieve the session keys in exchange. M_o can access, intercept or modify any data in transmission, but cannot decrypt any cipher text without the corresponding key or secret keying material; a cloud inside attacker M_i not only has the same ability as M_o, but also can be authenticated by the cloud as a legitimate server instance. However, M_i has no access to other instances or controllers or any of their secret keys. M_i aims to steal data of other users of the same cloud.

We gave a security proof to CCBKE in [22], and had the following theorems:

Theorem 1: A *cloud outside attacker* M_o cannot retrieve any exchanged session key g^{xy_i} in CCBKE in polynomial time.

Theorem 2: Assume $k = g^{xy_m}$ is the session key negotiated between a *cloud inside attacker* M_i and CLC. M_i cannot retrieve in polynomial time any session key g^{xy_i} other than k, unless a negligible probability in CCBKE.
Proofs: See [22].

Derived from these theorems, we now have the following lemma:

Lemma 1: The adversaries defined above have negligible chance of breaking the CBHKE scheme. Specifically, a cloud outside attacker M_o cannot retrieve any session key, while a cloud inside attacker M_i cannot retrieve any session key other than her/his own.
Proof: The key exchange procedures for each node and its sub-nodes in both CBHKE Phase 1 and Phase 2 are actually minimised and iterative CCBKE processes. As CCBKE is secure against cloud inside and outside attackers according to Theorems 1 and 2, all the KE operations in CBHKE scheme are secure against these attackers. Therefore, all the encrypted messages in our CBHKE scheme are securely encrypted. Hence, we can say that our new CBHKE scheme is secure against attackers from either outside or inside the cloud, defined in *Definition 1*.

In addition, if we use different parameters and keying materials for every execution and re-keying in CBHKE scheme, it will also hold *perfect forward security* just the same as in CCBKE and IKE.

E. Efficiency Analysis:

As analysed in section II, the majority of time consumption is from modular exponentiations, e.g., $g^x \bmod p$. Compared to them, the symmetric-key encryptions and decryptions in phase 2 take virtually no time because those concatenated keying materials to be encrypted are only several KBs long. Hence, we will analyse the efficiency advantage of our new CBHKE scheme by calculating the total number of modular exponentiations.

Let

$$M_i = \max_{1 \le j \le n_i} \{m_{i,j}\}$$

be the maximum number of sub-nodes for each node on level i. Starting from $M_1 = n_2$, we have

$$n_i = \sum_{j=1}^{n_{i-1}} m_{i-1,j} \le n_{i-1} M_{i-1}, i = 2, \dots, l$$

then the total number of VM instances is $\sum_{j=1}^{n_l} m_{l,j}$, with at most M_l VMs controlled by one NC. Assume the maximum time consumption of one modular exponentiations on one node is t_{max}, then the total time consumption of CCBKE is close to $\left(\sum_{j=1}^{n_l} m_{l,j}\right) t_{max}$ given that VM hold similar computational ability. In CBHKE, the upper bound of the total time consumption in KE modular exponentiations in one round should be $\left(\sum_{i=1}^{l} M_i\right) t_{max}$. Given the fact that each NC can launch and control plenty of VMs (much more than the number of control nodes controlled by a higher-level control node), the following inequality will hold:

$$M_l > \sum_{i=1}^{l-1} M_i$$

which is the case of both of our experimental environments. Besides, because we have $l \ge 2$ (otherwise CBHKE will have the exact same efficiency as CCBKE), we will have

$$\sum_{j=1}^{n_l} m_{l,j} \ge 2M_l$$

if the NCs have similar computational capability that can launch similar amount of VMs. Therefore:

$$\sum_{j=1}^{n_l} m_{l,j} \ge 2M_l > M_l + \left(\sum_{i=1}^{l-1} M_i\right) = \sum_{i=1}^{l} M_i$$

then we have

$$\left(\sum_{j=1}^{n_l} m_{l,j}\right) t_{max} > \left(\sum_{i=1}^{l} M_i\right) t_{max}$$

which means in practical cloud settings, CBHKE always has increased efficiency compared with CCBKE. In fact, in most cases we have:

$$M_l >> \sum_{i=1}^{l-1} M_i$$

then

$$\sum_{j=1}^{n_l} m_{l,j} \approx n_l \sum_{i=1}^{l} M_i$$

In this case, the time consumption of CBHKE is even only a fraction of CCBKE. Although IKE, CBHKE and CCBKE are all of linear time complexity to the scale of the task, the efficiency advantage of CBHKE is nonetheless tremendous.

V. EXPERIMENT AND EVALUATION

A. Experiment Environment

We conducted our experiments on U-Cloud (see Fig 2) -- a cloud computing environment located in University

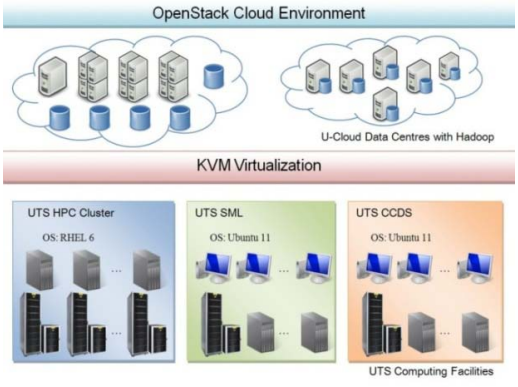

Fig. 2 U-Cloud structure

of Technology, Sydney (UTS). The computing facilities of this system are located in several labs in the Faculty of Engineering and IT, UTS. On top of hardware and Linux OS, We installed KVM Hypervisor [3] which virtualises the infrastructure and allows it to provide unified computing and storage resources. Upon virtualised data centres, Hadoop [2] is installed to facilitate the MapReduce programming model and distributed file system. Moreover, we installed OpenStack open source cloud platform [5] which is responsible for global management, resource scheduling, task distribution and interaction with users.

B. *Experiment Process*

We tested our scheme under two differently structured cloud instantiations of U-Cloud. The first one have 3 control layers and 4 NCs in total, while the second one have 4 control layers and 6 NCs in total. The layouts of the two experimental cloud scenarios are shown in Figure 3.

We implemented CBHKE, CCBKE and IKE schemes using C++ with MIRACL[4] cryptography library, and ran on our U-Cloud environment. The numbers of instances launched by each NC are $5, 10, 15, \ldots, 50$. On each of the cloud scenario, we repeatedly ran each of the key exchange scheme 20 times to simulate a big computation task with 20 CLC-VM interactions.

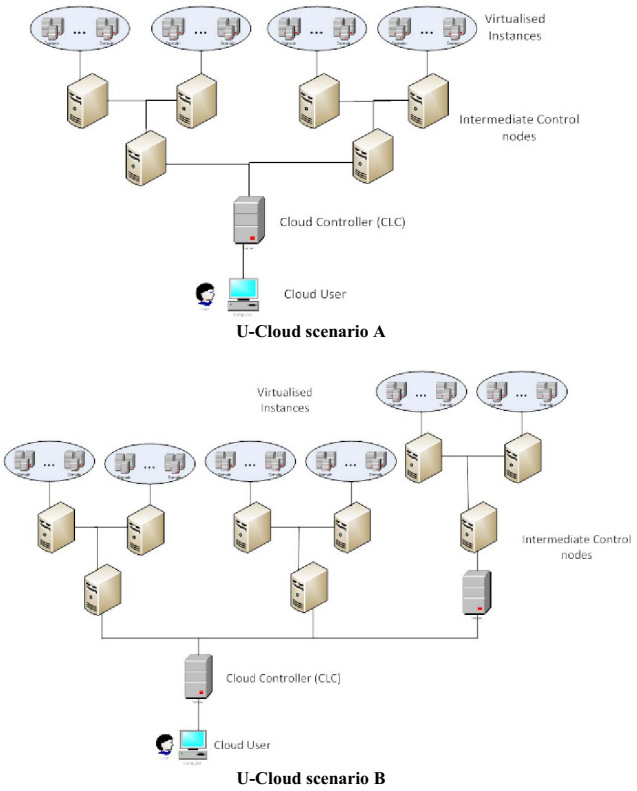

Fig.3 Structures of two cloud scenarios on U-Cloud for experiments

C. *Experiment Results*

Our experimental results (average for 20 runs) are shown in Figure 4. The results match the analysis in section IV - E. Through these results we can see that compared to IKE in U-Cloud, the total time consumption of CBHKE in KE is decreased by an average of 85.9% and 89.8% in scenarios A and B, respectively. This efficiency advantage of CBHKE when compared to CCBKE in the two scenarios is 70.96% (max: 75.9%; min: 58.9%) and 77.85% (max: 82.4%; min: 61.3%), respectively. This is a significant improvement in efficiency without compromising the level of security. Also, the results match our efficiency analysis in section IV.E.

VI. Conclusion and Future Work

In this paper we have proposed a novel hierarchical key exchange scheme, namely Cloud Background Hierarchical Key Exchange (CBHKE). Based on our previous work, CBHKE aimed at providing secure and more efficient scheduling for cloud computing environment. In our new scheme, we have designed a two-phase layer-by-layer iterative strategy to reduce the overall time consumption in authenticated key exchange (AKE) without sacrificing the level of data security. Both theoretical analysis and experimental results have demonstrated that when deployed in cloud computing environment, the proposed scheme was significantly much more efficient than its predecessors CCBKE and IKE.

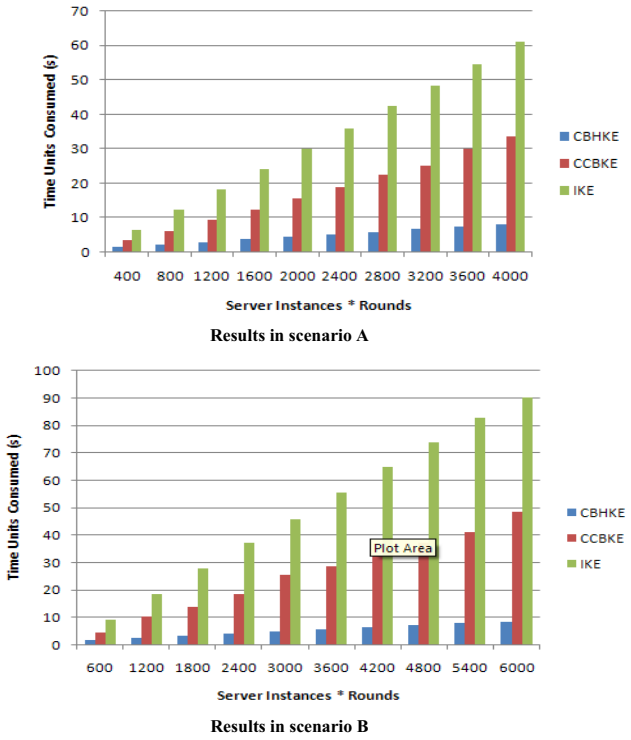

Fig.4 Efficiency comparison of CBHKE, CCBKE and IKE in the two cloud scenarios

Although the efficiency of security-aware scheduling in cloud computing can greatly benefit from an efficient key exchange (KE) scheme such as CBHKE, current encryption algorithms (even the fastest symmetric-key encryptions) are still not fast enough when encrypting large datasets, which is why currently another large research community is focusing on another way around -- to reduce encryption usage. Research on an efficient and secure encryption algorithm for cloud and other large-scale distributed computing systems is still an open research problem.

ACKNOWLEDGMENTS

This research work is partly supported by Australian Research Council under Linkage Project LP0990393.

REFERENCES

[1] *Australia Telescope, Parkes Observatory*. Available: http://www.parkes.atnf.csiro.au/, accessed on 10 March, 2013.

[2] *Hadoop MapReduce*. Available: http://hadoop.apache.org, accessed on 10 March, 2013.

[3] *KVM Hypervisor*. Available: www.linux-kvm.org/, accessed on 10 March, 2013.

[4] *MIRACL Cryptography Library*. Available: http://certivox.com/index.php/solutions/miracl-crypto-sdk/, accessed on 10 March, 2013.

[5] *OpenStack Open Source Cloud Software*. Available: http://openstack.org/, accessed on 10 March, 2013.

[6] D. Agrawal, S. Das, and A. E. Abbadi, "Big Data and Cloud Computing: Current State and Future Opportunities," in *Proceedings of the 14th International Conference on Extending Database Technology (EDBT/ICDT '11)*, Uppsala, Sweden, 2011.

[7] M. Bellare, A. Boldyreva, and J. Staddon, "Randomness Re-use in Multi-recipient Encryption Schemeas " in *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography (PKC '03)*, Miami, USA, 2003.

[8] R. Canetti and H. Krawczyk, "Security Analysis of IKE's Signature-Based Key-Exchange Protocol," in *Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '02)*, Santa Barbara, USA, 2002, pp. 143-161.

[9] S. Chaudhuri, "What Next?: A Half-dozen Data Management Research Goals for Big Data and the Cloud," in *Proceedings of the 31st symposium on Principles of Database Systems (PODS '12)*, Scottsdale, Arizona, USA, 2012, pp. 1-4.

[10] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The Cost of Doing Science on the Cloud: the Montage Example," in *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing (SC '08)*, Austin, Texas, 2008, pp. 1–12.

[11] S. K. Garg, C. S. Yeo, A. Anandasivam, and R. Buyya, "Environment-conscious Scheduling of HPC Applications on Distributed Cloud-oriented Data Centers," *Journal of Parallel and Distributed Computing,* vol. 71, pp. 732-749, 2011.

[12] C. Gentry, S. Halevi, and N. P. Smart:, "Fully Homomorphic Encryption with Polylog Overhead," in *Proceedings of the 2012 International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT '12)*, Cambridge, UK, 2012, pp. 465-482.

[13] A. Groce and J. Katz, "A New Framework for Efficient Password-based Authenticated Key Exchange," in *Proceedings of the 17th ACM conference on Computer and communications security (CCS '10)*, Chicago, USA, 2010, pp. 516-525.

[14] N. Heath. *Cern: Cloud Computing Joins Hunt for Origins of the Universe*. Available: http://www.techrepublic.com/blog/european-technology/cern-cloud-computing-joins-hunt-for-origins-of-the-universe/262, accessed on 10 March, 2013.

[15] J. Katz and V. Vaikuntanathan, "Round-optimal Password-based Authenticated Key Exchange," in *Proceedings of the 8th conference on Theory of cryptography (TCC'11)*, Providence, USA, 2011, pp. 293-310.

[16] J. Katz and M. Yung, "Scalable Protocols for Authenticated Group Key Exchange," *Journal of Cryptology,* vol. 20, pp. 85 - 113, 2007.

[17] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen. Internet Key Exchange Protocol Version 2 (IKEv2). *The Internet Engineering Task Force Request for Comments (IETF RFC) 5996*, September 2010. Available: http://tools.ietf.org/html/rfc5996, accessed on 10 March, 2013.

[18] D. Kawata, R. Cen, and L. C. Ho, "Gravitational Stability of Circumnuclear Disks in Elliptical Galaxies," *The Astrophysical Journal* vol. 669(1), pp. 232-240, 2007.

[19] K. Keahey, R. Figueiredo, J. Fortes, T. Freeman, and M. Tsugawa, "Science Clouds: Early Experiences in Cloud Computing for Scientific Applications," in *Proceedings of the First Workshop on Cloud Computing and its Applications (CCA '08)*, Chicago, USA, 2008, pp. 1 - 6.

[20] K. Kurosawa, "Multi-recipient Public-Key Encryption with Shortened Ciphertext," in *Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems: Public Key Cryptography (PKC '02)*, Paris, France, 2002, pp. 321–336.

[21] R. Küsters and M. Tuengerthal, "Computational Soundness for Key Exchange Protocols with Symmetric Encryption," in *Proceedings of the 16th ACM conference on Computer and communications security (CCS '09)*, Chicago, USA, 2009, pp. 91-100.

[22] C. Liu, X. Zhang, C. Yang, and J. Chen, "CCBKE - Session Key Negotiation for Fast and Secure Scheduling of Scientific Applications in Cloud Computing," *Future Generation Computer Systems,* to appear, 2012. doi: 10.1016/j.future.2012.07.001.

[23] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The Eucalyptus Open-Source Cloud-Computing System," in *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID '09)*, 2009, pp. 124-131.

[24] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, You, Get off of My Cloud: Exploring Information Leakage in Third-party Compute Clouds," in *Proceedings of the 16th ACM conference on Computer and communications security (CCS '09)*, Chicago, USA, 2009, pp. 199-212.

[25] X. Tang, L. Kenli, Z. Zeng, and B. Veeravalli, "A Novel Security-Driven Scheduling Algorithm for Precedence-Constrained Tasks in Heterogeneous Distributed Systems," *IEEE Transactions on Computers,* vol. 60, pp. 1017-1029, 2011.

[26] T. Xie and X. Qin, "Performance Evaluation of A New Scheduling Algorithm for Distributed Systems with Security Heterogeneity," *Journal of Parallel and Distributed Computing,* vol. 67, pp. 1067-1081, 2007.

[27] A. Y. Z. Young Choon Lee, "Energy Conscious Scheduling for Distributed Computing Systems under Different Operating Conditions," *IEEE Transactions on Parallel and Distributed Systems,* vol. 22, pp. 1374-1381, 2011.

[28] D. Yuan, Y. Yang, X. Liu, and J. Chen, "On-demand Minimum Cost Benchmarking for Intermediate Dataset Storage in Scientific Cloud Workflow Systems," *Journal of Parallel and Distributed Computing,* vol. 71, pp. 316-332, 2011.

[29] J. Zhao and D. Gu, "Provably Secure Authenticated Key Exchange Protocol under the CDH Assumption," *Journal of Systems and Software,* vol. 83, pp. 2297-2304, 2010.

[30] Z. Zhou and D. Huang, "An Optimal Key Distribution Scheme for Secure Multicast Group Communication," in *Proceedings of the 2010 IEEE Conference on Computer Communications (INFOCOM '10)*, San Diego, USA, 2010.