Enhancing Trusted Cloud Computing Platform for Infrastructure as a Service

Heeyoul KIM
Department of Computer Science, Kyonggi University, Suwon, 443760, Republic of Korea
heeyoul.kim@kgu.ac.kr

***Abstract*—The characteristics of cloud computing including on-demand self-service, resource pooling, and rapid elasticity have made it grow in popularity. However, security concerns still obstruct widespread adoption of cloud computing in the industry. Especially, security risks related to virtual machine make cloud users worry about exposure of their private data in IaaS environment. In this paper, we propose an enhanced trusted cloud computing platform to provide confidentiality and integrity of the user's data and computation. The presented platform provides secure and efficient virtual machine management protocols not only to protect against eavesdropping and tampering during transfer but also to guarantee the virtual machine is hosted only on the trusted cloud nodes against inside attackers. The protocols utilize both symmetric key operations and public key operations together with efficient node authentication model, hence both the computational cost for cryptographic operations and the communication steps are significantly reduced. As a result, the simulation shows the performance of the proposed platform is approximately doubled compared to the previous platforms. The proposed platform eliminates cloud users' worry above by providing confidentiality and integrity of their private data with better performance, and thus it contributes to wider industry adoption of cloud computing.**

***Index Terms*—authentication, communication system security, cryptographic protocols, data security, platform virtualization.**

I. INTRODUCTION

Cloud computing allows users to outsource storage and computing power on demand. NIST divides the services provided by the cloud computing into three categories [1]: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). IaaS provides the user with the capability to provision processing, storage, networks, and other computing resources where the consumer is able to deploy and run arbitrary software. Cloud computing is empowered by virtualization technology which provides the essential cloud characteristics of location independence, resource pooling and rapid elasticity [2]. In IaaS, the resources are provided in the form of virtualized systems, and several virtual machines are mapped to the same physical resources allowing the resource pooling in multi-tenant environment. A virtual machine monitor (VMM) or hypervisor manages the virtual machines and permits various operation systems to run simultaneously [3].

As cloud computing has been adopted widely, the importance of security in cloud computing has been emphasized [4]. The data in the cloud are more vulnerable in comparison to the conventional computing model, in terms of confidentiality, integrity and availability [5]. In the conventional model, the digital assets are controlled in the administrative domain of the owner's organization and trust is enforced according to own security policy. However, in the case of public or community clouds, administrative control is delegated to the organization owning the cloud service and infrastructure [6]. This mitigated control causes new risks and security issues. Moreover, not only the data at rest but also the data being processed cause security risks [7]. Sharing physical resources among multiple tenants may allow malicious users to launch attacks on the data of other users while in processing phase [8].

The security issue in managing virtual machines (VMs) is one of main concerns of cloud computing [9]. The virtual machines running on the same cloud node need to be isolated from each other. However, although they are logically isolated, the access to the same physical resources may lead to data breach. For example, VM escape [10] makes the attacker gain access to the hypervisor control over VMs running on the same node. This kind of threat significantly increases if a malicious employee in the cloud provider tries to attack. An inside attacker having administrative rights on a cloud node can access the memory of VMs running on the node to expose user's data. He also can compromise the hypervisor to put all the VMs managed by it under his control. During migration between cloud nodes via network, not only the contents of the VM can be exposed to the attacker, but also the code of the VM can become vulnerable [11]. Moreover, the attacker can lead the target VM to be relocated to a compromised hypervisor under his control.

The goal of this paper is to solve the security issue above in an efficient way. From the security aspect, providing cloud users both confidentiality and integrity of their data and computations eliminates the users' concern. From the performance aspect, efficiency in VM management increases the users' satisfaction. In this paper we present an enhanced trusted cloud computing platform which ensures the user's VM to be executed only on trusted cloud nodes against inside attackers. To guarantee both confidentiality and integrity of the VM, the platform includes improved protocols for VM management. The node registration protocol ensures only trusted cloud nodes are candidates for hosting the user's VM. The VM launch protocol ensures the initial VM is securely launched for hosting without being inspected or tampered. The VM migration protocol ensures the VM state is securely transferred between trusted nodes for migration. These protocols utilize both symmetric key operations and public key operations together with a simple

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the ministry of Education, Science and Technology (2011-0013757).

Digital Object Identifier 10.4316/AECE.2017.01002

and efficient node authentication model, whereas the previous platforms [12-13] excessively use public key operations with a complicated authentication model. This approach improves performance significantly by reducing both the computational cost and the communication steps.

The rest of this paper is organized as follows. Section 2 briefly describes related works. Section 3 provides the detailed design of the proposed platform. In Section 4, security analysis of the proposed platform is provided. In Section 5, both performance analysis and experimental result are provided. Finally, Section 6 concludes the paper.

II. Related Works

In order to provide reliability in computing devices, the Trusted Computing Group has developed the trusted platform module (TPM) which is mounted on the board acting as a trust anchor [14]. The TPM chip has a unique endorsement private key (EK) generated by a manufacturer and Platform Configuration Registers (PCRs) to assure integrity of the platform. The PCRs store a measurement list (ML) including a sequence of hashes of the software involved in the boot. Validating the ML assures that the device was not modified maliciously and it behaves as intended. Remote attestation [15], which allows changes in the host to be detected remotely by authorized parties, is provided by using of TPM chips in several trusted platforms.

Terra [16], one of trusted platforms, provides a trusted virtual machine monitor (TVMM) that partitions a single tamper-resistant platform into multiple isolated virtual machines. The TVMM enforces a closed-box execution environment and it protects privacy and integrity of the VM's contents. However, this platform can be operated only on a single node and it is not appropriate to cloud computing environment where many physical nodes are involved.

Infrastructure as a Service (IaaS) supports large numbers of VMs to be executed with abstracting users from the detail of infrastructure. By extending the concept of trusted platform to an entire IaaS backend, the trusted cloud computing platform (TCCP) was proposed [12]. It guarantees the confidentiality and the integrity of a user's VM, and it allows the user to attest to the IaaS provider. Each node in the IaaS backend embeds a TPM chip and securely installs the TVMM. The user's VM is protected from inside attacker's inspection or modification by using of the TVMM. These trusted nodes are managed by a trusted coordinator (TC) with several protocols including node registration protocol and VM launch protocol.

Several researches have been proposed to improve the TCCP. Trusted Eucalyptus [13], based on the Eucalyptus platform, has a very similar architecture to the TCCP and it utilizes an Attestation Identity Key (AIK) instead of EK for remote attestation to preserve privacy. Wang Han-zhang *et al*. [17] proposed an improved model in order to reduce too much dependency on the trusted third party. Ge Cheng *et al*. [18] proposed a TCCP with sealed storage ability which enables the users to bind sensitive data with their desired cloud environment integrity. Pojage *et al*. [19] proposed a distributed TCCP to avoid a single point of failure.

In the TCCP and its variations, most of the computations for the protocols are composed of public key operations which are much slower than symmetric key operations. Moreover, the mutual authentication between trusted nodes in the VM migration protocol is very complicated, requiring multiple protocol steps. These limitations decrease the overall performance of the platform, and consequently decrease cloud user's satisfaction also. On the other hand, the proposed platform in the next section overcomes these limitations and provides an efficient way without decreasing security level.

III. Proposed Platform

Here we present an enhanced trusted cloud computing platform which can guarantee both confidentiality and integrity of cloud user's virtual machine in IaaS environments. The overall architecture is similar to the TCCP, but we focus on the protocols for node and VM management. The TCCP and other existing platforms use public key operations excessively with complicated authentication model, which causes the limitations described above. The presented platform includes newly designed protocols to solve the limitations. The key idea is to design hybrid protocols that utilize both the symmetric key operations and the public key operations properly without harming the security level. The authentication model used in our protocols is also newly designed with the motivation from the Kerberos model [20].

The presented platform consists of the following main components, and Fig. 1 shows the overall architecture of the platform.

- **Cloud Manager (CM)**: Though a cloud management system has a set of components to manage cloud services, we assume a cloud manager takes charge of the role for simplicity. The cloud manager is administrated by the cloud provider, and we consider that it may be compromised by attackers and it may perform malicious behaviors.
- **Node (N)**: The node N in the cloud backend runs a TVMM that monitors multiple VMs in a closed-box execution environment. Each node embedding a TPM chip must process a secure booting before installing the TVMM. The node is considered as a trusted node after registration approved by the trusted coordinator below.
- **Trusted Coordinator (TC)**: The trusted coordinator handles trusted nodes to run the cloud user's VM securely. The TC attests to the node in the cloud provider to check whether it runs a proper TVMM satisfying security requirements. The TC also plays a key role in the VM management protocols including VM launch and VM migration. We assume the TC is a trusted third party outside the cloud and the cloud provider cannot manipulate the TC.

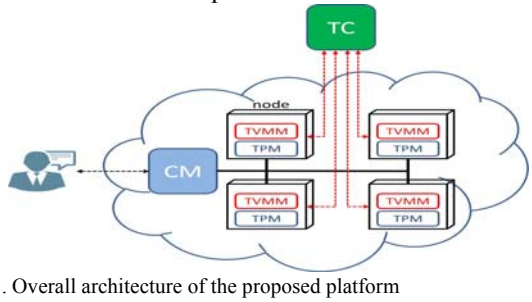

Figure 1. Overall architecture of the proposed platform

A trusted node running the TVMM can host a user's VM securely and can prevent it from inspection and modification by inside attackers. However, in the cloud the VM image is often transferred between nodes for launch or migration, and it can be exposed to the attackers while being transferred though the communicating nodes are trusted. In addition, the attacker can lead the VM to be launched or migrated in a compromised node. To overcome these risks, the presented platform provides several protocols. The following notations in Table I are used in these protocols.

TABLE I. NOTATIONS

Notation	Description
$< K_X^{pub}, K_X^{prv} >$	Public key and private key pair of X
$\{m\}_K$	Encryption of a message m with the key K
EK_X	Endorsement key of X
ML_X	Measurement list of X
TK_{TC}	Trusted key of TC
SK_X	Symmetric session key of X
n_X	Random nonce of X
$\#X$	Hash value of X

A. Node registration protocol

Among the nodes in the cloud provider, only trusted nodes having been assured by the TC must host the user's VM. This verification process is performed through the node registration protocol below. In the setup of the platform, for each candidate node N, the TC stores both the node's public endorsement key EK_N^{pub} and the expected measurement list ML_N for attesting to the node. In addition, the TC publicly opens its public endorsement key EK_{TC}^{pub} and its measurement list ML_{TC} for being attested by the nodes. The TC generates a key pair $< TK_{TC}^{pub}, TK_{TC}^{prv} >$, and opens TK_{TC}^{pub} to the nodes. A node N wanting to be registered performs the following protocol with the TC.

1) $N \rightarrow TC: n_N$

The node sends a random nonce to the TC to prevent replay attacks.

2) $TC \rightarrow N: \{ML_{TC}, n_N\}_{EK_{TC}^{prv}}, n_{TC}$

After verifying freshness of the nonce, the TC signs both the nonce and the measurement list with its private endorsement key. Then the TC sends it with a new nonce n_{TC}. The node attests to the TC by both verifying the signature with EK_{TC}^{pub} and checking the nonce.

3) $N \rightarrow TC: \{\{ML_N, n_{TC}\}_{EK_N^{prv}}, SK_N\}_{TK_{TC}^{pub}}$

The node signs both the nonce received and its measurement list with its private endorsement key for remote attestation. Then the node generates a session key SK_N. Both the signature and the session key are encrypted with the TC's public key to prevent eavesdropping, and they are sent to the TC. After receiving the message, The TC decrypts it and obtains both the signature and SK_N. Then the TC attests to the node by verifying the signature. If the ML_N matches the expected configuration, the TC adds the node in the trusted nodes list with SK_N.

4) $TC \rightarrow N: \{accepted\}_{SK_N}$

The TC sends a confirmation message to the node.

In comparison to the TCCP, the node generates and sends a symmetric session key SK_N to the TC instead of generating a trusted public key pair for the node. This key is utilized for both authentication and confidentiality in the VM launch protocol and VM migration protocol below.

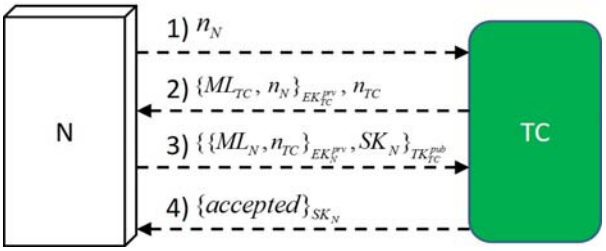

Figure 2. Node registration protocol

B. Virtual machine launch protocol

When a user u requests to launch a virtual machine in the cloud, the CM chooses a node from the candidates and forwards the request to the node for hosting. At this time the user cannot guess which node is assigned to host the image actually. However, the user expects that one of trusted nodes hosts the VM securely and that the inside attacker cannot inspect or tamper with the initial VM state while being transferred for launch. As he trusts only the TC among the platform components, the TC is responsible for authenticating the hosting node on behalf of the user.

1) $u \rightarrow N: \{\alpha, \#\alpha\}_{K_{VM}}, \{n_u, K_{VM}\}_{TK_{TC}^{pub}}$

The user generates a secret key K_{VM}, and encrypts both an initial VM image α and its hash value $\#\alpha$ with the key. Then he encrypts a nonce n_u and K_{VM} with the TC's public trusted key. These encrypted messages are sent to the CM. Then the CM chooses a node N for hosting and forwards received messages to N.

2) $N \rightarrow TC: \{n_u, K_{VM}\}_{TK_{TC}^{pub}}, \{n_N, N\}_{SK_N}$

The node N generates a nonce n_N and encrypts both the nonce and its identity with the session key SK_N. Then it sends the ciphertext with the message received in step 1) to the TC. The TC authenticates the node by decrypting $\{n_N, N\}_{SK_N}$ and then by checking the identity in the trusted nodes list. Then the TC obtains K_{VM} by decrypting the former message with its private trusted key TK_{TC}^{prv}.

3) $TC \rightarrow N: \{n_N, n_u, K_{VM}\}_{SK_N}$

The TC encrypts both the two nonces and the key K_{VM} obtained in step 2) with the session key SK_N. Then the message is sent to the node N. Now N can obtain the key K_{VM} by using of its session key, and it can then get the VM image α from the message in step 1).

4) $N \rightarrow: \{n_u, N\}_{K_{VM}}$

The node informs its identity to the user through the CM by sending a message encrypted with the key K_{VM}. Finally, the node boots the virtual machine.

In comparison to the TCCP, the key SK_N plays an important role in this protocol. In step 2), it is the basis for authenticating the node by the TC. In step 3), it is utilized to securely send K_{VM} for VM decryption to the node.

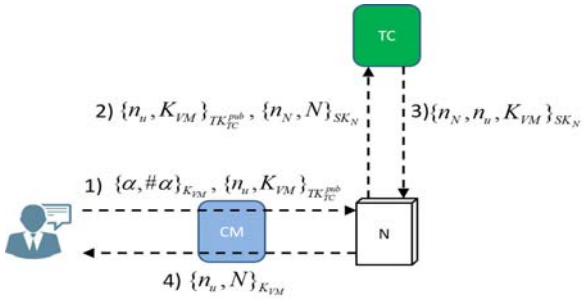

Figure 3. Virtual machine launch protocol

C. *Virtual machine migration protocol*

Live migration is utilized for load balancing and optimization of VM deployment in the cloud computing. When a node gets overloaded, some of the VMs on it are moved to other nodes with minimal interruption to the users. Actually moving the VM means transferring VM state including memory and CPU state. Let us assume a live migration from a source node N_s to a destination node N_d. To protect the user's data in the VM from insider attackers, not only the nodes should be trusted but also the VM state should not be exposed during transfer. The following protocol satisfies these requirements. This protocol is totally different from the one in the TCCP because authentication of the two nodes are based on the Kerberos model.

1) $N_s \rightarrow TC : \{N_d, N_s, n_{s1}, n_{s2}\}_{SK_{N_s}}$

When the node N_s receives an order from the CM to migrate a VM to the node N_d, it encrypts the identity of itself, the identity of the destination node, and two random nonces with the session key SK_{N_s}. After receiving the encrypted message, the TC authenticates N_s and finds the destination node by decrypting the message. The TC also checks that both the nodes are in the trusted nodes list.

2) $TC \rightarrow N_s : \{N_s, K, n_{s2}\}_{SK_{N_d}}, \{n_{s1}, K\}_{SK_{N_s}}$

The TC generates a secret key K to be utilized for VM state encryption. Then the TC encrypts N_s, K, and the nonce n_{s2} with the session key SK_{N_d} shared only with N_d. Also the TC encrypts n_{s1} and K with the session key SK_{N_s}. The two encrypted messages are sent to N_s. Only the latter message can be decrypted by N_s, and the former message will be forwarded to N_d in the next step.

3) $N_s \rightarrow N_d : \{N_s, K, n_{s2}\}_{SK_{N_d}}, \{n_{s2}, N_s\}_K$

After obtaining the key K, N_s encrypts both n_{s2} and N_s with K for being authenticated by N_d. Then it is sent to the destination node N_d with the message $\{N_s, K, n_{s2}\}_{SK_{N_d}}$ received in step 2). After receiving them, N_d can decrypt the former message with its session key and thus can obtain the key K. Then N_d decrypts the latter message with K and checks both the identity and the nonce.

4) $N_d \rightarrow N_s : \{n_{s2} - 1\}_K$

N_d sends a message $n_{s2} - 1$ encrypted with the key K. Then N_d is authenticated by checking the value.

5) $N_s \rightarrow N_d : \{\alpha, \#\alpha\}_K$

N_s computes the hash value of current VM state α to be migrated. Then both the VM state and its hash value are encrypted with the secret key K only shared with N_d. The encrypted message sent to N_d provides the VM state to be hosted after checking the hash value.

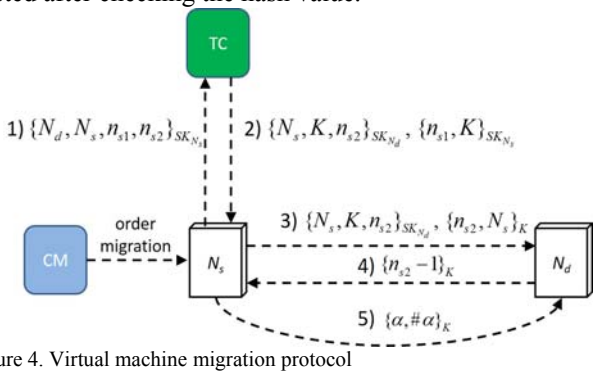

Figure 4. Virtual machine migration protocol

The presented platform solves cloud users' worry about exposure of their private data in IaaS environment. The platform provides both confidentiality and integrity of their data and computations by guaranteeing the VM is hosted on only trusted nodes and by preventing the VM is exposed over the network. Also, the platform increases cloud users' satisfaction by improving performance with the proposed VM management protocols. The detailed analyses of both security and performance are presented in section 4 and in section 5 respectively.

IV. Security Analysis

The main worry for cloud users using IaaS is that their secret information and computation in their virtual machines are exposed to the attackers. The conventional computing model has focused on preventing it from outside attacks, however, in cloud computing the users cannot trust even the internal node consisting of the cloud service. The presented platform provides the way to solve this concern. Here we assume the attacker can be an employee of the cloud provider and he can remotely access to cloud nodes with administrative privilege for trying to get sensitive information. We do not assume the attacker can physically access to the nodes for trying some side channel attacks such as cold boot attack.

The presented platform not only guarantees the VM is hosted on only trusted nodes, but also prevents the VM's exposure over the network during launch and migration. The public key encryption and the symmetric key encryption used in the protocols provide confidentiality of both the VM state and transferred messages, and the hash value provides integrity of the VM state against tampering trials. The TCCP and its variations also provide confidentiality and integrity of the running VMs. However, they utilize only public key operations except for VM encryption. In addition, the public key is treated in a different way than the conventional concept. In the node registration protocol, each node generates a public-private key pair and the public key is sent to the TC securely instead of being opened to the public. If a node wants to authenticate another node, it has to get the opponent's public key by asking the TC. Moreover, because there is no mechanism such as digital certificates to verify

current validity of the public key, it has to ask the TC again for the public key whenever authentication is needed. This makes the protocols (especially node migration protocol) complicated without additional security functionality. The presented platform, on the other hand, provides simple protocols by utilizing both public key and symmetric key.

The cloud nodes are treated as trusted only after being authenticated and being attested by the TC in the node registration protocol. The remote attestation in this protocol guarantees that the node provides proper execution environment and especially that the TVMM is working correctly to host the virtual machine. The TVMM makes even the administrator of the node cannot access into the virtual machine running on it. Therefore, the users trusting the TC are assured that their VMs are running securely.

Through the registration a trusted node shares a secret session key with the TC. All nodes consider the TC as a trusted third party, and each node never shares the key with any other nodes. Thus sharing this symmetric key instead of using public key pair does not threaten the platform. And this key plays an essential role in the virtual machine launch protocol. Because the user does not know on which node the virtual machine will be launched, he sends the secret key K_{VM} in the form only the TC can decrypt and others cannot. Then the node forwards it with proving that the node itself is a trusted node by using of the session key. After successful authentication, the TC securely send the node K_{VM} for decrypting the VM. This process ensures that the user's VM was delivered to one of trusted nodes and only the node can read the VM image.

The VM migration protocol is more complicated than the others because both the source node N_s and the destination node N_d must authenticate each other and also must verify the other side is a trusted node. To provide this in a secure and efficient way, our platform proposes an authentication mechanism motivated from the Kerberos system [20]. At first N_s requests authentication for migration. The TC verifies both nodes are trusted, and authenticates N_s. Then it issues a ticket $\{N_s, K, n_{s2}\}_{SK_{N_d}}$ proving that migration between N_s and N_d has been granted. This ticket is sent to N_s with a newly generated secret key K. The source node N_s can authenticate the destination node by checking the response after forwarding the ticket. For the correct response, the destination node should discover both K and n_{s2} by decrypting the ticket received. Only the legitimate node N_d knowing the session key SK_{N_d} can discover them. Meanwhile, the destination node N_d can authenticate the source node after receiving messages in step 3). N_d decrypts the ticket to get the key K, and consequently it decrypts the message $\{n_{s2}, N_s\}_K$ with K to check both the nonce n_{s2} and the identity N_s. The ticket means N_s has been authenticated by the TC as a trusted node. And only the node N_s knowing both n_{s2} and K can make the correct message $\{n_{s2}, N_s\}_K$. Hence N_d is convinced that the source node N_s is a trusted node.

V. Performance Analysis and Experiment

Performance of cloud services is one of key factors. Cloud users expect improved performance thanks to the elastic computing where computing resources can be scaled up and down easily by the cloud service provider. However, elasticity consumes some overhead obviously. It takes a certain time for a virtual machine to be ready to run when it is launched or migrated. As the users lose confidence in the service provider due to poor performance and delayed response time, performing efficient VM launch and migration helps to increase the users' satisfaction.

The previous platforms including the TCCP and the Trusted Eucalyptus excessively use public key operations that are relatively very slow. This design choice increases the execution time of the protocols. Whereas, the presented platform provides improved protocols where both public key operations and symmetric key operations are utilized together with the efficient authentication model explained. This approach reduces the execution time of the protocols, and consequently improves the overall performance. Our claim is further clarified by both the cost analysis and the simulation result below.

Table II shows the comparison of the number of protocol steps and cryptographic operations with the previous ones. The proposed platform reduces the number of public key encryptions and signings at the expense of symmetric key encryptions. Especially this difference is emphasized in the migration protocol where the presented platform requires no public key operations and 6 symmetric key encryptions. This improves performance significantly as it is obvious that public key operation is hundreds or a thousand times slower than symmetric key operation depending on the implementation techniques environment.

TABLE II. COMPARISON OF PROTOCOL STEPS AND CRYPTOGRAPHIC OPERATIONS

platform		# of protocol steps	# of public key encryption	# of private key signing	# of symmetric key encryption
TCCP [12]	registration	4	2	2	0
	launch	4	3	2	2
	migration	7	5	5	2
Trusted Eucalyptus [13]	registration	4	2	2	0
	launch	4	3	2	2
	migration	5	2	3	2
Proposed	registration	4	1	2	1
	launch	4	1	0	4
	migration	5	0	0	6

We also have simulated both the presented platform and the TCCP for the comparison, and it is focused on estimating the amount of time to perform the protocols. A small-sized cloud was composed of trusted nodes with 3.1 GHz CPU and 4GB RAM. Both the user client and the TC were set up physically away from the cloud. In our simulation, the protocols were implemented in Java language, and Jersey framework was utilized for HTTP based communication. AES algorithm with 128-bit key was chosen for symmetric key operation, and RSA algorithm with 1024-bit key was chosen for public key operation.

As the node registration protocol in this platform has little difference with the one in the TCCP, we omitted it in the simulation. We assume the nodes have been registered and the session keys were already assigned to the nodes before initiating simulation. We used a small-sized dummy block instead of a real virtual machine image to focus on the protocol performance itself. We measured the elapsed time of the launch protocol from the user's request to the arrival of the message in step 4). We measured the elapsed time of the migration protocol from the CM's migration order to the destination node's obtainment of the image.

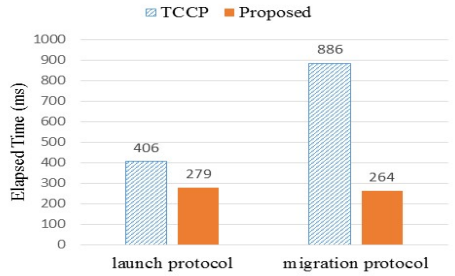

Figure 5. Comparison of average time of launch and migration protocols

We simulated these two protocols a hundred times separately, and Fig. 5 shows the average time measured in milliseconds. The presented platform reduces the elapsed time of the launch protocol by roughly 30%. Moreover, in case of the migration protocol, the elapsed time is drastically reduced by roughly 70% due to not only reduced cryptographic operations but also reduced communication steps. This result shows that our platform based on hybrid protocols improves performance significantly.

VI. Conclusion

Cloud users worry their credential is exposed through the virtual machines running in IaaS environment by inside attackers. In this paper, we have proposed a trusted and efficient cloud platform to overcome this concern. The platform utilizes the TPM and remote attestation to validate trusted cloud nodes, and the platform presents hybrid protocols to provide node registration, virtual machine's secure launch and migration. The protocols utilize both symmetric key operations and public key operations together for both security and efficiency. In addition, a new authentication model is presented for improving efficiency.

The contribution of this platform in comparison with the existing platforms is that it solves the security issue above in an efficient way. It reduces cryptographic computation time significantly by removing most public key operations at the expense of additional symmetric key operations, and it reduces communication steps in the migration protocol. Thus it has better performance than the previous platforms. The experiment also shows that the presented platform improves performance of the protocols. In the aspect of security, this platform also guarantees both confidentiality and integrity of cloud user's virtual machine in IaaS environments. This contribution is especially emphasized in large-scale cloud services, and it is expected to increase cloud users' satisfaction.

References

[1] P. M. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-145, 2011.

[2] M. Armbrust, I. Stoica, M. Zaharia, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, and A. Rabkin, "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, p. 50, Apr. 2010. doi:10.1145/1721654.1721672

[3] T. Kaur and I. Chana, "Energy Efficiency Techniques in Cloud Computing: A Survey and Taxonomy," ACM Computing Surveys, vol. 48, no. 2, pp. 1–46, Oct. 2015. doi:10.1145/2742488

[4] M. Ali, S. U. Khan, and A. V. Vasilakos, "Security in cloud computing: Opportunities and challenges," Information Sciences, vol. 305, pp. 357–383, Jun. 2015. doi:10.1016/j.ins.2015.01.025

[5] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, "Toward Secure and Dependable Storage Services in Cloud Computing," IEEE Transactions on Services Computing, vol. 5, no. 2, pp. 220–232, Apr. 2012. doi:10.1109/TSC.2011.24

[6] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," Future Generation Computer Systems, vol. 28, no. 3, pp. 583–592, Mar. 2012. doi:10.1016/j.future.2010.12.006

[7] M. Sookhak, H. Talebian, E. Ahmed, A. Gani, and M. K. Khan, "A review on remote data auditing in single cloud server: Taxonomy and open issues," Journal of Network and Computer Applications, vol. 43, pp. 121–141, Aug. 2014. doi:10.1016/j.jnca.2014.04.011

[8] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," Journal of Internet Services and Applications, vol. 4, no. 1, p. 5, 2013. doi:10.1186/1869-0238-4-5

[9] K. Ren, C. Wang, and Q. Wang, "Security Challenges for the Public Cloud," IEEE Internet Computing, vol. 16, no. 1, pp. 69–73, Jan. 2012. doi:10.1109/MIC.2012.14

[10] M. H. Song, "Analysis of Risks for Virtualization Technology," Applied Mechanics and Materials, vol. 539, pp. 374–377, Jul. 2014. doi:10.4028/www.scientific.net/AMM.539.374

[11] F. Zhang and H. Chen, "Security-Preserving Live Migration of Virtual Machines in the Cloud," Journal of Network and Systems Management, vol. 21, no. 4, pp. 562–587, Dec. 2013. doi:10.1007/s10922-012-9253-1

[12] N. Santos, K. P. Gummadi, and R. Rodrigues, "Towards trusted cloud computing," Proc. HotCloud'09, Article no. 3, 2009.

[13] I. Khan, H. Rehman, and Z. Anwar, "Design and Deployment of a Trusted Eucalyptus Cloud," Proc. IEEE cloud computing, 2011, pp. 380–387. doi:10.1109/cloud.2011.105

[14] S. Balfe, A. D. Lakhani, and K. G. Paterson, "Trusted Computing: Providing Security for Peer-to-Peer Networks," Proc. IEEE PSP'05, pp. 117–124. doi:10.1109/p2p.2005.40

[15] D. G. Murray, G. Milos, and S. Hand, "Improving Xen security through disaggregation," Proc. VEE'08, 2008, p. 151. doi:10.1145/1346256.1346278

[16] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh, "Terra: a virtual machine-based platform for trusted computing," ACM SIGOPS Operating Systems Review, vol. 37, no. 5, p. 193, Dec. 2003. doi:10.1145/1165389.945464

[17] Wang Han-Zhang and Huang Liu-Sheng, "An improved trusted cloud computing platform model based on DAA and privacy CA scheme," Proc. ICCAMS 2010, 2010, pp. V13-33-V13-39. doi: 10.1109/ICCASM.2010.5622643

[18] Ge Cheng and A. K. Ohoussou, "Sealed storage for trusted cloud computing," Proc. ICCDA 2010, 2010, pp. V5-335-V5-339. doi:10.1109/iccda.2010.5541060

[19] S. R. Pojage and M. A. Pund, "Review of trusted cloud computing platform security," Proc. NCSC2D 2016, pp. 167-172, Feb. 2016.

[20] B. C. Neuman and T. Ts'o, "Kerberos: an authentication service for computer networks," IEEE Communications Magazine, vol. 32, no. 9, pp. 33–38, Sep. 1994. doi:10.1109/35.312841