Trusted Cloud Computing Platform into Infrastructure as a Service Layer to Improve Confidentiality and Integrity of VMs

Divyesh Yoganand
PG student
Computer Science &Engineering
NRI Institute of Information Science& Technology, Bhopal, India

Pooja Kose
Professor
Computer Science &Engineering
NRI Institute of Information Science& Technology, Bhopal, India

ABSTRACT
Out of the newly emerging and promising technologies is Cloud computing and Infrastructure-as-a-Service (IaaS) which can also be claimed as something the adoption of which is hampered by data security concerns. Simultaneously, Trusted Computing (TC) is also getting its burning interest as security mechanism for IaaS. This paper presents a protocol and addresses the issue of the lack of an implementable mechanism with a proportion that it will ensure the launch of a virtual machine (VM) instance on a trusted remote compute host. A trusted launch protocol for VM instances and images in public IaaS environments has been designed for Relying on Trusted Platform Module operations such as binding and sealing to provide integrity guarantees for clients that require a trusted VM launch. This paper also presents an evidence -of-concept implementation of the protocol that is solely based on Open Stack, an open-source IaaS platform. The proposed results would provide a strong stand for the use of TC mechanisms within IaaS platforms. It will also open the path for a bigger applicability of TC to IaaS security. This technology empowers the companies to take the costs down by outsourcing computations which are on-burning demand. Nevertheless, clients of cloud computing services at present do not have any means by which they can verify the confidentiality and integrity of their data and computation. This problem is addressed to propose the design of a trusted cloud computing platform (TCCP). To impart a closed box execution environment, TCCP empowers Infrastructure as a Service (IaaS) providers such as Open stack IaaS platform. It also ensures the confidential execution of guest virtual machines. Besides, it also lets the users confirm to the IaaS provider and determine if the service is secure before they launch their virtual machines.

General Terms
Security, TCCP protocol, Open stack IAAS platform

Keywords
IaaS, security, trusted computing, trusted virtual machine launch, OpenStack, Cloud Computing, Scalability, Infrastructure, confidentiality, integrity, trusted cloud computing platform.

1. INTRODUCTION
One of the distinguished trends in IT operations in the present era is the consolidation of IT systems onto common platforms. The core technology in this is system provides room for streamline IT operations, save energy and obtains better utilization of hardware resources. It permits the users to run own services in form of Virtual Machines (VM) on shared computing resources. This approach however introduces new challenges, as it means that information previously controlled by one administrative domain and organization, is now under the control of a third party provider and that the information owner loses direct control over how data and services are used and protected. IaaS[4] is one of the business models based on system virtualization and security aspects are among the main identified obstacles for its adoption of IAAS. The problems with securing IaaS are evident not least through the fact that widely known platforms such as Amazon EC2, Microsoft Azure, services provided by Rackspace and other IaaS services are plagued by vulnerabilities at several levels of the software stack, from the web based cloud management console [5] to VM side-channel attacks, to information leakage, to collocation with malicious virtual machine instances. [6]

To prevent confidentiality violations, cloud services' customers might route to encryption. While encryption is effective in securing data before it is stored at the provider, it cannot be applied in services where data is to be computed, since the unencrypted data must reside in the memory of the host running the computation. In Infrastructure as a Service (IaaS) cloud services such as Amazon's EC2 and Open stack IaaS platform the provider hosts virtual machines (VMs) on behalf of its customers, who can do arbitrary computations. In such systems, anybody with privileged access to the host can see or manipulate a customer's data. As a result, customers cannot protect their VMs on their own. Considerable effort to secure their systems is made in order to minimize the threat of insider attacks, and reinforce the confidence of customers. For example, access to the hardware facilities are restricted and protected, adopt stringent accountability and auditing procedures, and minimize the number of staff who has access to dangerous components of the infrastructure. Nevertheless, insiders that administer the software systems at the provider backend ultimately still possess the technical means to access customers' VMs. Thus, there is a clear need for a technical solution that ensures the confidentiality and integrity of computation thus that is verifiable by the customers of the service.

This paper proposes a trusted cloud computing platform (TCCP) for ensuring the confidentiality and integrity of computations that are outsourced to IaaS services. The TCCP provides the abstraction of a closed box execution environment for a customer's VM, guaranteeing that no cloud provider's privileged administrator can inspect or tamper with its content. Moreover, before requesting the service to launch a VM, the TCCP allows a customer to reliably and remotely determine whether the service backend is running a trusted TCCP implementation. This capability extends the notion of attestation to the entire service, and thus allows a customer to verify if its computation will run securely.

2. 2. BACKGROUND

2.1 Infrastructure as a Service

Today, myriads of cloud providers offer services at various layers of the software stack. At lower layers, Infrastructure as a Service (IaaS) providers such as Amazon, Flexiscale, and GoGrid allow their customers to have access to entire virtual machines (VMs) hosted by the provider. A customer, and user of the system, is responsible for providing the entire software stack running inside a VM. At higher layers, Software as a Service (SaaS) systems such as Google Apps offer complete online applications than can be directly executed by their users.

The difficulty is in ensuring the confidentiality of computations increases for services sitting on higher layers of the software pile because services themselves impart and run the software that directly manipulates customer's data (e.g., Google Docs). In this paper it is focused that the lower layer IaaS cloud providers where securing a customer's VM is more manageable.

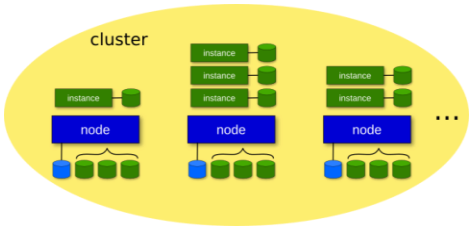

Figure 1: Simplified Architecture of IaaS

2.2 Trust and attack models

In present IaaS providers, it can be reasonably considered that no single person accumulates all these privileges. Moreover, providers already happen to arrange stringent security devices, restricted access control policies, and surveillance mechanisms to protect the physical integrity of the hardware. In this way, researcher presumes that, by enforcing a security perimeter, the provider itself can prevent attacks that require physical access to the machines. Though, sysadmins require privileged permissions at the cluster's machines to manage the software they run. Since we do not exactly know the praxis of current IaaS providers, we assume in their attack model that sysadmins can login remotely to any machine with root privileges, at any point in time. The only way a sysadmin would be able to gain physical access to a node running a costumer's VM is by diverting this VM to a machine under her control, located outside the IaaS's security perimeter. Therefore, the TCCP must be able to 1) confine the VM execution inside the perimeter, and 2) guarantee that at any point a sysadmin with root privileges remotely logged to a machine hosting a VM cannot access its memory.

Researcher shares the attack model with [11,12,13] which considers that privileged access right scan be maliciously used by remote system administrators (Ar) of theIaaS provider. In current IaaS providers, it can be reasonably considered that no single person accumulates all these privileges. Moreover, providers already organize stringent security devices, restricted access control policies, and surveillance mechanisms to protect the physical integrity of the hardware. Thus, researcher assumes that, by enforcing a security perimeter, the provider itself can prevent attacks that require physical access to the machines.

2.3 Trusted Computing

The Trusted Computing Group (TCG) [10] proposed a set of hardware and software technologies to enable the construction of trusted platforms. In particular, the TCG proposed a standard for the design of the trusted platform module (TPM) chip that is now bundled with commodity hardware. The TPM holds an endorsement private key (EK) that can distinctively identify the TPM (thus, the physical host), and some cryptographic functions which can never be modified. The respective manufacturers sign the corresponding public key to guarantee the correctness and validity of the key.

Trusted platforms [7, 8, 9, 10] leverage the features of TPM chips to enable remote attestation. This mechanism works as follows. At boot time, the host computes a measurement list ML consisting of a sequence of hashes of the software involved in the boot sequence, namely the BIOS, the boot loader, and the software implementing the platform. The ML is securely stored inside the host's TPM. To confirm to the platform, a remote party challenges the platform running at the host with a nonce nU. The local TPM is asked to create a message containing both the ML and the nU, encrypted with the TPM's private EK. The host sends the message back to the remote party who can decrypt it using the EK's corresponding public key, thereby authenticating the host. By checking that the nonces match and the ML corresponds to a configuration it deems trusted, a remote party can reliably identify the platform on an untrusted host. A trusted platform like Terra implements a thin VMM that enforces a closed box execution environment, meaningthat a guest VM running on top cannot be examined or modified by a user with full privileges over the host. The VMM assures its integrity till the machine does not reboot. Thus, a remote party can attest to the platform running at the host to verify that a trusted VMM implementation is running, and thus make it sure that her computation running in a guest VM is secured. Given that a traditional trusted platform can secure the computation on a single host, a natural approach to secure an IaaS service would be to deploy the platform at each node of the service's backend (see Figure 1).

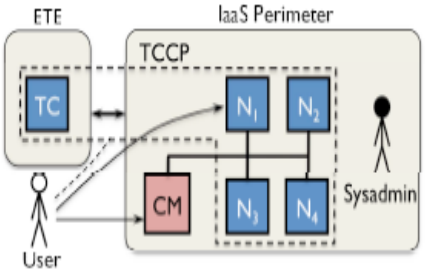

Figure 2: The components of the trusted clocomputingPlatform include a set of trusted nodes (N) and the trusted coordinator (TC). The entrusted cloud manager(CM) makes a set of services available to users. The TCis maintained by an external trusted entity (ETE).

3. TRUSTED CLOUD COMPUTING PLATFORM

Researcher launch the trusted cloud computing platform (TCCP) that provides a closed box execution environment by extending the concept of trusted platform to an entire IaaS backend. The TCCP guarantees the confidentiality and the

integrity of a user's VM, and allows a user to determine up front whether or not the IaaS enforces these properties.

3.1 Overview

TCCP enhances today's IaaS backends to enable closed box semantics without substantially changing the architecture (Figure 2). The trusted computing base of the TCCP includes two components: a trusted virtual machine monitor (TVMM), and a trusted coordinator (TC).

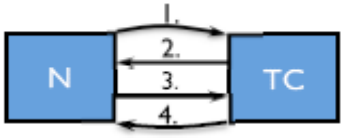

Figure 3: Message exchange during node registration.

Each node of the backend runs a TVMM that hosts customers' VMs, and prevents privileged users from inspecting or modifying them. Over time, the integrity of TVMM is protected by it and complies with the TCCP protocols. Nodes embed a certified TPM chip and must go through a secure boot process to install the TVMM. Due to space limitations researcher need not go into detail about the design of the TVMM, and researcher refer the reader to [14] for an architecture that can be leveraged and followed to build a TVMM that enforces local closed box protection against a malicious sysadmin.

1. n_N

2. $\{ML_{TC}, n_N\}EK^P_{TC}, n_{TC}$

3. $\left\{\{ML_N, n_{TC}\}EK^P_N, TK^P_N\right\}_{TK^P_{TC}}$

4. $\{accepted\}_{TK^P_N}$

The TC manages the set of nodes that can run a customer's VM securely. It is called trusted nodes. To be trusted, a node

must be located within the security perimeter, and run the TVMM. To meet these conditions, the TC maintains a record of the nodes located in the security perimeter, and attests to the node's platform to verify that the node is running a trusted TVMM implementation. It is said that the TC can cope with the occurrence of events such as adding or removing nodes from a cluster, or shutting down nodes temporarily for maintenance or upgrades. A user can verify whether the IaaS service secures its computation by attesting to the TC.

To secure the VMs, each TVMM running at each node cooperates with the TC in order to 1) confine the execution of a VM to a trusted node, and to 2) protect the VM state against inspection or modification when it is in transit on the network. The critical moments that require such protections are the operations to launch, and migrate VMs. Researcher hypothesize that an external trusted entity (ETE) that hosts the TC, and securely updates the information provided to the TC about the set of nodes deployed within the IaaS perimeter among others. Above all, sysadmins that manage the IaaS have no privileges inside the ETE, and therefore cannot tamper with the TC. Researcher envisions that the ETE should bemaintained by a third party with little or no incentive to collude with the IaaS provider e.g., byindependent companies analogous to today's certificate authorities like VeriSign.

3.2 Detailed Design

In this section, researcher provides the details the most relevant TCCP mechanisms. Researcher narrate the protocols that manage the set of nodes of the platform that are trusted (Section 3.2.1), and the protocols that secure the operations involving VM management, namely launching and migrating VMs (Section 3.2.2). In these protocols, the following notation for cryptographic operations is used. The pair hKp, KP it represents the private-public keys of an asymmetric cryptography key pair. Notation {y}Kx indicates that data y is encrypted with key Kx. Researcher use a specific notation for the following keys: EKx denote endorsement keys, TKx indicate trusted keys, and Kx denote session keys. Nonces nx, unique numbers generated by x, help detect message replays.

3.2.1 Node management

The TC dynamically operates the set of trusted nodes that can host a VM by maintaining a directory containing, for each node within the security perimeter, the public endorsement key EKP N identifying the node's TPM, and the expected measurement list MLN. The ETE makes some properties of the TC securely available to the public, namely the EKP TC, the MLTC, and the TKP TC (identifying the TC). The canonical configurations are expressed by both the MLN and the MLTC that a remote party is expected to observe when attesting to the platform running on a node N or on the TC, respectively.

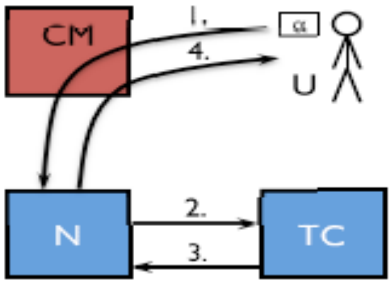

Figure 4: Message exchange during VM launch

1. $\{\alpha, \#\alpha\}K_{VM}\{n_U, K_{VM}\}_{TK^P_{TC}}$

2. $\left\{\left\{\{n_U, K_{VM}\}_{TK^P_{TC}}, n_N\right\}_{TK^P_N}, N\right\}_{TK^P_{TC}}$

3. $\left\{\{n_N, n_U, K_{VM}\}_{TK^P_N}\right\}_{TK^P_{TC}}$

4. $\{n_U, N\}_{K_{VM}}$

In order to be trusted, a lump must register with the TC by complying with the protocol depicted on Figure 3. In steps 1 and 2, N attests to the TC to avoid an impersonation of the TC by an attacker: N sends a challenge nN to the TC, and the TC replies with its bootstrap measurements MLTC encrypted with EKp TC to guarantee the authenticity of the TC. If the MTC matches the expected configuration, it means the TC is trusted. Reversely, the TC also attests to N by piggybacking a challenge nTC in message 2, and checking whether the node is authentic, and is running the expected configuration (step 3). The node generates a key pair <hTKpN, TKPN> and sends its public key to the TC. If

both peers mutually attest successfully, the TC adds TKPN to its node database, and sends message 4 to confirm that the node is trusted. Key TKN ensures that node N is trusted. A trusted node reboots, theTCCP must assure that the node's configuration remains trusted; or the node could compromise the security of the TCCP. To ensure this, the node onlykeeps TKp N inmemory causing the key to be lost once the machine reboots. The node is thus banned from the TCCP, since it will not be able to decrypt messages encrypted with the previous key, and must repeat the registration protocol.The first purpose of TCCP is to ensure the confidentiality and integrity of cloud consumer's computations. it is based on trusted open stack is used in their prototype implementation. Per 5 minutes analysis and statistics are shown in figure 5.

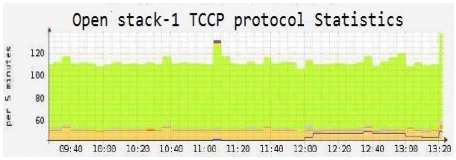

Figure 5: TCCP Protocol Statistics

3.2.2 Virtual machine management

Researcher represent the TCCP protocols to secure the VM launch and migration operations. When launching a VM, the TCCP needs to guarantee that 1) the VM is launched on a trusted node, and 2) the sysadmin is unable to inspect or tamper with the initial VMstate as it traverses the pathbetween the user and the node hosting the VM. The initial VM state contains the VM image (VMI) (that can be personalized and contain secret data) and the user's public key (used for ssh login) 1. In practice, the user can decide to use a VMI provided by the IaaS.

To implement these requirements, the parties involved in launching a VM follows the protocol depicted in Figure 4.This protocol has been designed on the fact that, before launching the VM, a user does not know which physical node the VM will be assigned, and, among the components ofthe service, only trusts the TC. First, the user generates a session key KVM, and sends message 1 to the CM containing: α and α's hash encrypted with the session key (to protect the confidentiality and integrity of the initial state), and KVM encrypted with TKP TC. Encrypting the session key with the TC's public key ensures that only the TC can authorize someone to access α. The TC only authorizes trusted nodes.

After receiving the request to launch a VM, the CM designates a node N from the cluster to host the VM, and forwards the request to N. Since the node needs to access α in order to boot the VM, it sends message 2 to TC which decrypts KVM on N's behalf. This message is encrypted with TKp N so that the TC can verify whether N is trusted. If the corresponding public key is not found in the TC's trusted node database, the request is denied. This would have been the case had the CM diverted the request to a node controlled by a malicious sysadmin. Otherwise, the node is reckoned to be trusted; the TC decrypts the session key, and sends it to the node in message 3, such that only N can read the key. N is now able to decrypt α, and boot the VM. Finally, message 4 is sent by the node to the user containing the identity of the node running the VM.

In live migration [3], the state of an executing VM is transferred between two nodes: a source Ns anda destination Nd. To secure this operation, both nodes must be trusted, and the VM state must remain confidential and unmodified while it is in transit over the network. Figure 4 shows the sequence of messages involved in securing the migration of a VM. In steps 1 and 2, Ns asksTC to check whether Nd is trusted. In message 3, Ns negotiates a session key KS with Nd that will be used to secure the transfer of the VM state.Before accepting the key, Nd first verifies that Ns is trusted (steps 4 and 5). If both nodes mutually authenticate successfully, Nd acknowledges the acceptance of the session key to the KS (step 6), and, in message 7, Ns finally transfers the encrypted and hashed VM state to the Nd, guaranteeing the confidentiality and integrity of the VM.

4. PROTOCOL IMPLEMENTATION

4.1 Open stack IaaS platform

The Essex release of OpenStack comprises _ve core components (projects), namely Compute (Nova), Image Service (Glance), Object Storage (Swift), Identity Service (Keystone) and Dashboard (Horizon). Nova has several sub-components: nova-api, nova-compute, nova-schedule, nova-network, nova-volume, plus an SQL database and message queue functionality to pass messages between sub-components. OpenStack components affected by the protocol implementation are mentioned here in more detail:

-Nova-api is the interface for nova- compute and volume API calls. It is through this interface most of the cloud orchestration operations are performed. The interface supports both the OpenStack and Amazon EC2 APIs.

-Nova-compute handles virtual machine instance life cycle tasks through hypervisor API calls. Notably the libvirt and XenAPI hypervisor APIs are supported.

-Nova-schedule is responsible for selecting compute host(s) to run virtual machine instances on. The host selection process is determined by which scheduling policy/algorithm is employed.

-The nova SQL database holds tables and relations to describe the state of nova, such as launched instances and network configurations.

-The Dashboard is a web based GUI for OpenStack operation and administration. It interfaces nova-api.

5. CONCLUSIONS AND FUTURE WORK

Thus it is argued that concerns about the confidentiality and integrity of their data and computation are a major deterrent for enterprises looking to embrace cloud computing. It is presented in this design of a trusted cloud computing platform (TCCP) that enables IaaS services such as Amazon EC2 to provide a closed box execution environment. TCCP assures confidential execution of guest VMs, and allows users to attest to the IaaS provider and determine if the service is secure before they launch their VMs. It is planned to implement a fully functional prototype based on their design and evaluate its performance in the near future. Moreover, researcher has provided a prototype implementation of the launch protocol in Open Stack. The given results make a case for broadening the range and also limits of use cases for trusted computing by applying it to IaaS environments, in particular within the security model of an entrusted IaaS provider.

6. REFERENCES

[1] Nicolae Paladi1, Christian Gehrmann1, Mudassar Aslam1, and Fredric Morenius2. "Trusted Launch of Virtual Machine Instances in Public IaaS Environments" October 2011, AFCEA cyber communit.

[2] Nicolae Paladi1, Christian Gehrmann1, Mudassar Aslam1, and Fredric Morenius2. "Trusted Launch of Virtual Machine Image in Public IaaS Environments" October 2011, AFCEA cyber communit.

[3] Nuno Santos, Krishna P. Gummadi, Rodrigo Rodrigues: Towards Trusted Cloud Computing (MPI-SWS)

[4] Aryan Taherimonfared Securing IaaS services model of cloud computing against compromised components 2011.

[5] Somorovsky,J Heiderich,M., Jensen,M., Schwenk, J., Gruschka,N., LoIacono, L.: All Your Clouds Are Belong to us: Security Analysis of Cloud Management Interfaces.In: Proceedings of the 3rd ACM Workshopon Cloud ComputingSecurity.CCSW'11,NewYork, NY,USA,ACM(2011) 3–14

[6] Ristenpart, T., Tromer, E., Shacham, H., Savage, S.: Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds. In: Proceedings of the 16th ACM Conference onComputer and Communications Security. CCS '09, New York, NY, USA, ACM (2009) 199–212

[7] S. Berger, R. C´aceres, K. A. Goldman, R. Perez, R. Sailer, and L. van Doorn. vTPM: virtualizing the trusted platform module. In Proc. of USENIX-SS'06, Berkeley, CA, USA, 2006.

[8] Survey: Cloud Computing 'No Hype', But Fear of Security and Control Slowing Adoption.http://www.circleid.com/posts/20090226_cloud_computing_hype_security/.

[9] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. In Proc. of NSDI'05, pages 273–286, Berkeley, CA, USA, 2005.USENIX Association.

[10] Santos,N.,Gummadi,K.P.,Rodrigues,R.:Towards Trusted Cloud Computing.In: Proceedingsofthe2009Conference onHot Topics in Cloud Computing.HotCloud'09,Berkeley,CA,USA,USENIX Association(2009)

[11] Aslam, M.,Gehrmann,C., Rasmusson,L., Bjorkman,M.:Securely LaunchingVirtual Machines on TrustworthyPlatforms inPublicCloud-An Enterprise'sPerspective. In Leymann,F., Ivanov, I., van Sinderen,M., Shan, Teds.: CLOSER,SciTePress(2012)51

[12] Aslam,M.,Gehrmann,C.,Bjorkman,M.:Security andTrust Preserving VM Migration sin Public Clouds. In:2012IEEE 11thInternationalConference on Trust, Securit and Privacy in Computingand Communications(TrustCom), TRUSTCOM,Liverpool (2012)

[13] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. Terra: A Virtual Machine-Based Platform for Trusted Computing. In Proc. of SOSP'03, 2003D. G. Murray, G. Milos, and S. Hand. Improving Xen security through disaggregation. In Proc. of VEE'08, pages 151–160, NewYork, NY, USA, 2008.

[14] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman,L. Youseff, and D. Zagorodnov. Eucalyptus: A Technical Report on an Elastic Utility Computing Architecture Linking Your Programs to Useful Systems. Technical Report 2008-10, UCSBComputer Science, 2008.

[15] B. D. Payne, M. Carbone, and W. Lee. Secure and Flexible Monitoring of Virtual Machines. In Proc. of ACSAC'07, 2007.

[16] T. R. Peltier, J. Peltier, and J. Blackley. Information Security Fundamentals. Auerbach Publications, Boston, MA, USA, 2003.

[17] R. Sailer, T. Jaeger, E. Valdez, R. Caceres, R. Perez, S. Berger, J. L. Griffin, and L. v. Doorn. Building a MAC-Based Security Architecture for the Xen Open-Source Hypervisor. In Proc. Of ACSAC '05, Washington, DC, USA, 2005

[18] Smith, J., Nair, R.: Virtual Machines: Versatile Platforms for Systems and Processes.Mor-ganKaufmann(June2005)

[19] Krutz,R.L., Vines, R.D.: Cloud Security:A Comprehensive Guide to Secure Cloud Com-puting.John Wiley &Sons(August2010)