A survey on Improvement of virtual network communication security of trusted launch of virtual machine in public IAAS environment

Divyesh Yoganand [1] Pooja Kose[2]
[1]M. Tech.Student [2]Assistant Professor
[1,2]Department of Computer Science & Engineering
[1,2]NRI Institute of Information Science & Technology, Bhopal, India

Abstract— Cloud computing and Infrastructure-as-a-Service (IaaS) are emerging and promising technologies, however their faster-pased adoption is hampered by data security concerns. At the same time, Trusted Computing (TC) is experiencing an increasing interest and revived interest as a security mechanism for IaaS. In this paper we present a protocol and We address the lack of an implementable mechanism to to ensure the launch of a virtual machine (VM) instance on a trusted remote compute host. Relying on Trusted Platform Module operations such as binding and sealing to provide integrity guarantees for clients that require a trusted VM launch, we have designed a trusted launch protocol for VM instances and images in public IaaS environments. We also present a proof-of-concept implementation of the protocol based on OpenStack, an open-source IaaS platform. The results provide a basis for the use of TC mechanisms within IaaS platforms and pave the way for a wider applicability of TC to IaaS security.

Keywords: IaaS, security, trusted computing, trusted virtual machine launch, OpenStack, Cloud Computing, Scalability, Infrastructure

I. INTRODUCTION

One of the distinguished trends in IT operations today is the consolidation of IT systems onto common platforms. A key technology in realizing this is system virtualization.[3] System virtualization makes it possible to streamline IT operations, save energy and obtain better utilization of hardware resources. A virtualized computing infrastructure allows clients 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]

A promising approach towards reducing IaaS security threats and a mean to pro-vide service con_dence is the use of Trusted Computing technologies as de_ned by the Trusted Computing Group (TCG) [7]. The core component in the TCG-de_ned secu-rity architecture is the Trusted Platform Module (TPM), a hardware module that can be used as a trust anchor for software integrity veri_cation in open platforms that also offers protected storage for sensitive parameters. TPM usage and deployment models for IaaS clouds are currently an active research area [,8,9,10,11,12,13]. Earlier research has introduced principles of a trusted IaaS platform [11], later extended to cover both trusted VM launch [12] and VM migration [13]. These research results demonstrate principles of combining basic TPM attestation mechanisms with standard cryptographic techniques to design an infrastructure for VM protection. However, such solutions have limitations with respect to security, complexity and target compute host selection procedures.

In this paper we describe a trusted VM launch process that addresses these limitations and present a trusted launch protocol that does not require secure pre-packaging of the VM image on the client side. The proposed protocol is in particular suitable for launching Generic virtual machine images (GVM images), i.e., VM images without any customer- Specific modifications Furthermore, in order to be usable in a significant proportion of IaaS deployment scenarios and to provide full scheduling exibility on the IaaS side, the protocol allows the IaaS provider to select a target trusted compute host without directly involving the client. The main contributions of this paper are:

(1) Description of a trusted launch protocol for VM instances in public IaaS environments.
(2) Implementation of the proposed protocol based on a widely-known IaaS platform.
(3) Description of a trusted launch protocol for generic VM images in IaaS environments.

II. LITERATURE SURVEY

Trust and attack models, problem description and requirements

A. Trust and attack models

We share the attack model with [11,12,13] which considers that privileged access rights can be maliciously used by remote system administrators (Ar) of the IaaS provider. This implies that we use a scenario which assumes that Ar can log in remotely to any host maintained by the IaaS provider and obtain root access. However, in this model Are does not have physical access to the hosts. The only possibility for Ar to circumvent this constraint is by succeeding to force a client to launch their VM instances on a host outside the physically secured IaaS provider perimeter and controlled by the Ar. Furthermore, we assume that an Ar obtaining remote root access to the host will not be able to directly access the memory of the VMs residing on the host at that

time, i.e. the cloud host platform offers a closed box execution environment.

(1) The VM image used for the instance is itself trusted;
(2) The VM instance is started on a trusted compute host;
(3) The VM instance has the client-generated verification token injected;

B. Virtual machine images

As an implication of the above trust and attack models, we consider the following two properties of virtual machines in the context of trusted computing:

- No VM instance, or any entity communicating with the VM instance, can determine whether the hypervisor the VM instance is running on is trusted or not.
- A VM instance cannot be trusted to reliably determine if it has the configuration originally requested by the client.

To overcome these issues, we suggest a launch protocol where we use standard TPM v1.2 functionality to first ensure that the client can detect the situation when it is communicating with a VM instance that is not launched on a trusted platform and subsequently utilize the trusted platform to verify the integrity of the VM image prior to VM launch. It is essential, in the scope of the protocol, that no modi_cations or customizations of the VM image to be launched are performed by the IaaS provider without the client's knowledge.

C. Generic virtual machine images

Denote by V the whole set of binary VM images and by v a particular VM image offered by a vendor. Furthermore, denote by vt the GVM image offered to an arbitrary client at time t:

Definition 1. $\forall v \in V, \forall t, t^0 \in T: \quad VT \equiv v_t{}^0$

To overcome these issues we suggest a launch protocol where we employ the TPM functionality to first make sure that the GVM image is actually launched on a trustworthy platform and subsequently utilize the trusted platform to verify the integrity of the GVM image prior to the VM launch. The protocol performs both steps while maintaining transparency from both client and IaaS provider's points of view.

D. Requirements for a trusted VM launch protocol and trusted GVM image launch protocol

Considering the trust and attack models above, it is important for the client to be able to obtain reasonable security guarantees from the IaaS provider. These include both trustworthiness of the computing resources, as well as guarantees regarding VM integrity and confidentiality. In order to also be cost and implementation efficient, the Underlying infrastructure should provide such guarantees with a minimal operational Overhead without increasing structural complexity. The expectations can be summarized as a set of basic requirements towards a trustworthy VM launch process:

R1: The client shall have the mechanisms to ensure that the VM instance and GVM image has been launched on a trusted compute host.

R2: The client should have the possibility to reliably determine that it is communicating with a VM instance and GVM image launched on a trusted compute host and secure host respectively not with a different VM instance and GVM instance.

R3: The integrity of the VM image and GVM image scheduled to be launched must be verifiable by the target trusted compute host and target computer host.

R4: The trusted VM launch and trusted GVM image procedure should be scalable and have a minimum impact on the performance of the IaaS platform.

R5: Clients should have a transparent view of the and secure trusted launch procedure.

III. A TRUSTED LAUNCH PROTOCOL FOR VIRTUAL MACHINE IMAGES AND A SECURE LAUNCH PROTOCOL FOR GENERIC VMS IN IAAS ENVIRONMENTS

The protocol requires the participation of four entities, three of which are typically involved in VM launch procedures in IaaS architectures:

(1) Client (C) is a IaaS user and intends to launch a VM instance. In this paper, C is considered to be a non-expert, i.e. one not capable of assessing the security of platform configurations based on values contained in the measurement list. C requires a VM instance to be launched on a trusted platform. Furthermore, it is important for C to be able to either verify or trust the security of VM images provided for launch.
(2) Scheduler (S) is responsible for receiving requests for VM instance launches from C, as well as scheduling and rescheduling of VM instances on available compute hosts at the IaaS provider. S should be able to function with minimal involvement in the security-specific message passing.
(3) The compute host (CH) is the target resource that will be chosen by S to run the particular VM instance. CH represents a physical or virtual server that is able to host one or more VM instances (however, this paper considers exclusively the case when the CH is a physical server). For the purposes of the proposed protocol, a CH must also be equipped with a TCG-compliant TPM as well as be immune to modification attempts when in a trusted state.
(4) The Trusted third party (TTP) is, as the name implies, trusted by both the Client and the IaaS provider and cannot be controlled or manipulated by the IaaS provider. The recent breaches of Certificate Authorities have emphasized the drawbacks of centralized security models and their susceptibility to attacks. The more complex the operations performed by the TTP, the higher the probability of it having exploitable vulnerabilities. It is therefore important to keep the implementation of the TTP as simple as possible. The main task of the TTP is to attest the configuration of the CH that will host the VM instance and assess its security profile according to predefined policies. Within the current trust model, TTPs could be implemented by

an expert C, as long as the IaaS provider agrees to that and C has the capability to set up and operate an attestation and evaluation engine.

IV. A TRUSTED LAUNCH PROTOCOL FOR VIRTUAL MACHINE IMAGES AND A SECURE LAUNCH PROTOCOL FOR GENERIC VMS PLATFORM-AGNOSTIC PROTOCOL DESCRIPTION

The following steps are required in order to perform a trusted VM launch and trusted generic VM launch

(1) Before initiating the launch procedure, client C generates a sufficiently long nonce N, to be used as a proof token in communications between the C and the VM instance and must be kept secret throughout the launch process.

(2) C creates a token which we denote by T , representing a data structure with information necessary for the trusted VM launch. T contains N, the preferred SP and the hash of the VM image to be launched, denoted as HVMimage. Finally, the token is encrypted with the public key of TTP, denoted as PKTTP , while the encrypted token is noted as TPKTTP.

(3) C requests the scheduler (S) to load a generic VM by providing the following parameters in the request:
{ VM type (e.g. CentOS, Debian, etc.);
{ Required SP;
{ URL of the T T P;
{ Encrypted token TPKTTP generated in step (2);
SP will determine the lower bound of trust level required from the host CH on which the VM will run, with stricter security pro_les accepted.

(4) S schedules a VM on the appropriate compute host, depending on its membership in the respective security pro_le group and sends a request to generate a bind key PKBind, also providing the URL of the TTP.

(5) Once the destination host CH receives the bind key request, it retrieves a PCR- locked non-migratable TPM-based bind key PKBind. This key can be periodically regenerated by CH according to a administrator-defined policy, using the current platform state represented by the TPM PCR. It is important to note that the values of the PCRs should not necessarily be in a trusted state in order to create a trusted state bind key.

(6) In order to prove that the bind key is a non-migratable, PCR-locked, asymmetric TPM key, CH uses the TPM CERTIFY KEY TPM command in order to retrieve the TPM CERTIFY INFO structure signed with the TPM attestation identity key [17], which we denote as PKAIK; we also denote the signed structure by HTPM CERTIFY INFO AIK. The TPM CERTIFY INFO data structure contains the hash of the bind key and the PCR value required for the key usage.

(7) CH sends an attestation request to the TTP through an HTTPS session using the URL supplied by the C. The following arguments are sent in the request to TTP:
{ Client-provided token TPKTTP
{ Attestation data, which includes the public bind key, the TPM CERTIFY INFO structure, the hash of TPM CERTIFY INFO signed with the AIK8, the IML and the AIK-certificate collectively represented as: PKBind; TPM CERTIFY INFO, HTPM CERTIFY INFO
AIK; IML, AIK-cert .

(8) TTP uses its private key PrKTTP , which corresponds to the public PKTTP to attempt to decrypt the token TPKTTP .

(9) T T P validates the attestation information obtained from CH as follows:
{ Validates the AIK certificate;
{ Validates the structure of the AIK-signed TPM CERTIFY INFO;
{ Validates the key PKBind by comparing its digest with the digest received in TPM CERTIFY INFO;
{ Calculates the hash of the PCR values HPCR based on the information in the IML and compares it with the hash of PCR INFO, which is a component of TPM CERTIFY INFO

(10) TTP examines the entries in the IML in order to determine the trustworthiness of the platform and decides whether the security preference SP is satisfied by the current configuration of compute host CH.

(11) If step 10 is true, TTP encrypts N and the hash HVMimage with the bind key PKBind obtained from CH, to ensure that N is only available to CH in a trusted state. By sending N encrypted with the public key PKBind available to the trusted configuration of CH, the security perimeter expands to include three parties: C itself, stateless T T P and compute host CH in its trusted configuration. This implies that all actions performed by CH in its trusted configuration are trusted by default.

(12) Prior to launching the VM, compute host CH decrypts N and HVMimage using the TPM-issued PrKBind, which is available to it in its trusted con_guration but stored in the TPM; next, CH compares HVMimage obtained from the TTP with the hash of the VM image offered by the IaaS provider and accepts the image for launch only in case the values are equal.

(13) CH injects N into the VM image prior to launching the VM.

(14) CH returns an acknowledgement to S to confirm a successful launch.

(15) To verify that the requested VM image has been launched on a secure platform, C challenges the VM launched on host CH to prove its knowledge of N.

V. PROTOCOL IMPLEMENTATION

A. Open Stack IaaS platform

The Essex release of OpenStack comprises five 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.

VI. RESEARCH FINDINGS

First is the extension of the trust chain to other operations on VM instances (migration, suspension, updates, etc.), as well as data storage and virtual network communication security.

The second category includes addressing certain assumptions of the proposed launch protocol, e.g. the assumption that the CH configuration is not changed after the trusted launch of the VM instance, since even in the case of a bona fide IaaS provider the CH can be compromised through runtime process infection. A technique to enable C to either directly or through mediated access discover such events or protect the data used by the VM instance is a promising research topic.

The third category focuses on the design and implementation of the evaluation policies of the TTP. The current assumption is that the TTP has access to information regarding \secure" configurations and the PCR values, something which needs to be directly addressed as evaluating exactly how secure a certain software stack is, is a challenge. Furthermore, taking into account the diversity of available libraries as well as the different combinations in which they can be loaded during the boot process, verification of PCR values (such as values stored in PCR10 and reference values in binary runtime measurements) becomes a less trivial task.

VII. RESEARCH SCOPE

I believe that using the proposed secure virtualization architecture, even under an untrusted management OS, a trusted computing environment can be created for a VM which needs a high security level, with very small performance penalties.

Our future research vision will focus on two directions to provide confidentiality, integrity, and secure infrastructure management for IaaS service. First, extending techniques such as proposed in TCCP into IaaS layer to improve confidentiality and integrity of VMs. Second, integrating TCCP with secure resources management schemes to get more controlled isolation environment. We also plan to flesh out the assessment framework further, supported by tools – to aid migration of enterprise applications to cloud.

We also plan to investigate the impact of VMs interferences in our resource management algorithms. My research questions will center on application and data security over the cloud, and Iintend to develop a framework by which the security methodology varies dynamically from one transaction/communication to another.

VIII. CONCLUSION

In this paper we have presented a detailed trusted launch protocol for VM instance and generic VM image launch in public IaaS environments. Furthermore, we have provided a prototype implementation of the launch protocol in Open Stack. The presented results make a case for broadening the range of use cases for trusted computing by applying it to IaaS environments, especially within the security model of an untrusted IaaS provider. Trusted computing offers capabilities to securely perform data manipulations on remote hardware owned and maintained by a third party with a minimal risk for data integrity and potentially preventing the use of untrusted software on that hardware for such manipulations The presented design is directly applicable to the process of developing a trusted virtualized environment within a public IaaS service.

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] Smith, J., Nair, R.: Virtual Machines: Versatile Platforms for Systems and Processes. Mor- gan Kaufmann (June 2005)
[4] Krutz, R.L., Vines, R.D.: Cloud Security: A Comprehensive Guide to Secure Cloud Com- puting. John Wiley & Sons (August 2010)
[5] Somorovsky, J., Heiderich, M., Jensen, M., Schwenk, J., Gruschka, N., Lo Iacono, L.: All Your Clouds Are Belong to us: Security Analysis of Cloud Management Interfaces. In: Proceedings of the 3rd ACM Workshop on Cloud Computing Security.CCSW ’11, New York, 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 on Computer and Communications Security. CCS ’09, New York, NY, USA, ACM (2009) 199–212
[7] Pohlmann, N., Reimer, H.: Trusted Computing - eine Einfhrung. In Pohlmann, N., Reimer, H.,

eds.:Trusted Computing. Vieweg+Teubner (2008)3–12
[8] Neisse, R., Holling, D., Pretschner, A.: Implementing Trust in Cloud Infrastructures. In: Cluster, Cloud and Grid Computing (CCGrid), 2011 11th IEEE/ACM International Sym- posium on. (may 2011) 524 –533
[9] Sadeghi, A.R., Stüble, C.,Winandy, M.: Property-Based TPM Virtualization. In Wu, T.C., Lei, C.L., Rijmen, V., Lee, D.T., eds.: Information Security. Volume 5222 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2008) 1–16
[10] Danev, B., Masti, R.J., Karame, G.O., Capkun, S.: Enabling Secure VM-vTPM Migration in Private Clouds. In: Proceedings of the 27th Annual Computer Security Applications Conference. ACSAC '11, New York, NY,USA, ACM (2011) 187–196
[11] Santos,N.,Gummadi, K.P., Rodrigues, R.:Towards Trusted Cloud Computing.In: Pro-ceedings of the 2009 Conference on Hot Topics in Cloud Computing.HotCloud'09, Berkeley, CA,USA,USENIX Association (2009)
[12] Aslam, M.,Gehrmann, C., Rasmusson, L., Bjorkman, M.: Securely Launching Virtual Machines on Trustworthy Platforms in a Public Cloud - An Enterprise's Perspective. In Leymann, F., Ivanov, I., van Sinderen, M., Shan, T eds.: CLOSER, SciTePress (2012)51
[13] Aslam, M., Gehrmann, C., Bjorkman, M.: Security and Trust Preserving VM Migrations in Public Clouds. In: 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom),TRUSTCOM, Liverpool (2012)
[14] Kuyoro S. O.,Ibikunle F.,Awodele O.- Cloud Computing Security Issues and Challenges(2011).

