Revision History

0.1	Added Sec. 1 Overview: presenting the attestation scenario.	
	Added Sec. 2 Three-layer Deep-Quote: illustrating attestation framework in our scenario.	
	Added Sec. 3 Implementation: added diagrams for software implementing the deep-quote	
0.2	Added known issues and (proposed) solutions to Sec. 3.2	

Remote Attestation in Eucalyptus

1. Overview

The goal of the mTC project is to give the Eucalyptus cloud users the capabilities of verifying the integrity of their Virtual Machines (VMs) running on the cloud. In this scenario, the Trusted Computing framework proposed by the Trusted Computing Group (TCG) can be integrated into Eucalyptus to provide remote attestation services to the users.

Considering the Eucalyptus infrastructure, as depicted in Figure 1, the integrity of a VM relies on three different parts. Attestations should be made towards them separately in order to provide proofs for integrity of the entire VM's lifecycle.

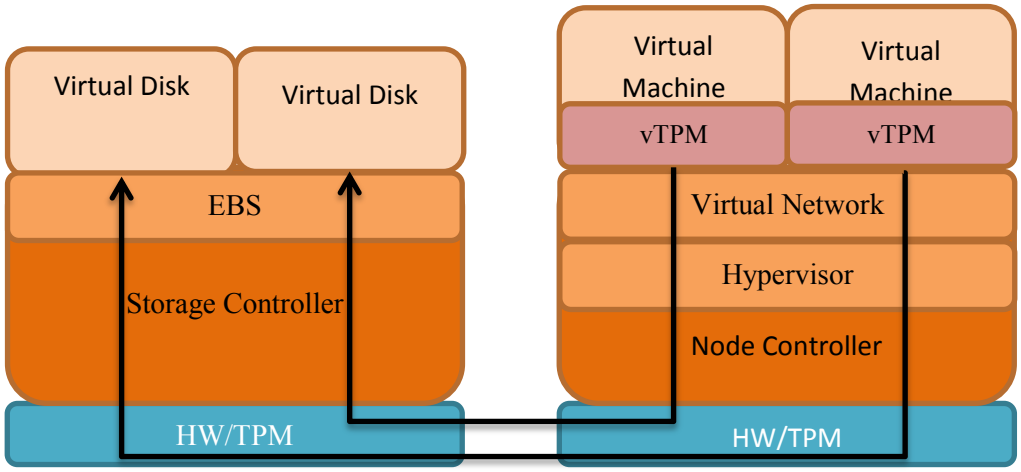

Figure 1. Logical layers of the dependencies of VMs on Eucalyptus

Attestation to Virtual Machines. Attestation at this scale can ensure that only expected programs with expected configuration files (including the data to process) are loaded inside the VM. It implements fine-grained attestation to the internal running state of a VM.

Attestation to Node Controllers. Attestation at this scale can ensure that the expected VM has been loaded on the cloud and its state can only be manipulated by genuine software stack. By "expected" we denote that the VM the user currently connecting to is genuinely loaded by the genuine hypervisor with the specified parameters, including the files for kernel images and root images and other configurations, such networking parameters and virtual storage binding parameters.

Attestation to Storage Controllers. Attestation at this scale can ensure that the VM is binding to the expected virtual storage, and the state of the virtual storage can only be manipulated by genuine software stack. By "expected" we denote that the virtual storage the user's VM currently connecting to is genuinely loaded and managed by the genuine EBS with the specified parameters.

2. Three-layer Deep-Quote

Attestations to these three components can be performed separately, i.e. users initiate three different attestation sessions to them, and then verify their integrity and combine them to form a general conclusion. However, the major defect of this attestation scheme is that the internal infrastructure of the cloud should be exposed to the users. Users should talk directly to the Node Controllers (NCs) and the Storage Controllers (SCs), widening the attack surface of the cloud. Attestations can also be delegated to a Trusted Third Party for verifying the integrity of those three components for every client's altogether. But this centralized attestation scheme may result in single-point-of-failure and is not easy to scale as the dimension of the cloud grows.

Instead, we can utilize a layered attestation scheme, which can be referred to as deep-quote. The users directly attest to the VMs they are connecting to, and the returned attestation tickets contain three parts, representing the integrity of the VM itself, the NC hosting it and the SC connecting to it.

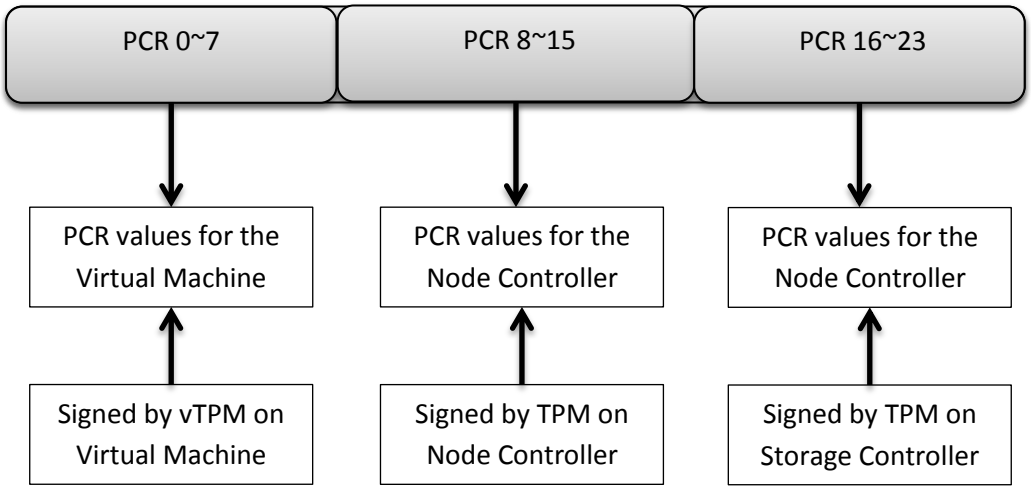

Figure 2. Attestation ticket for the three-layer deep-quote

Attestation. As depicted in Fig. 2, the attestation ticket returned by VM returns in responding to the user's attestation request contains three parts of PCR values. The first contains the PCR values recoded in the vTPM representing the integrity of the software stack inside the VM. It is signed by the AIK (Attestation Identity Key) of the vTPM. The second contains the PCR values of the underlying NC. It is fetched by the VM at the same time with the VM attestation. The same nonce for the VM attestation can be used to guarantee the freshness of these PCR values. These values are then signed by the AIK of the TPM on the NC. The third part contains the PCR values of the connected SC. It is fetched and signed in the same way as the NC attestation. The AIK of the TPM on the SC is used instead.

Binding. A nonce can guarantee the freshness of an attestation session, avoiding replay attacks. However, proofs of binding should also be provided, i.e. those three entities are not only genuine at the time of attestation but also actually working together. Otherwise a tampered NC or SC can perform an attestation to a genuine NC or SC with the provided nonce and return those responded tickets. To avoid this Man-In-The-Middle (MITM) attack, when the NC/SC receive a new attestation request from a VM,

additional checks should be made to examine whether the requesting VM is running on / connecting to them. As the genuine enforcement of this checking functionality can also be reflected in the PCR values, we assert that NC/SC with the expected PCR values can assure these strong bindings. Moreover, the PCR values of NC/SC can be bound with the PCR values of the VM, e.g. the NC/SC hash the PCR values of the VM and the nonce as the new nonce for their attestations. Hence the MITM in the VM layer can be avoided: redirecting the attestation request to a genuine VM on the same NC with the tampered VM.

3. Implementation

1) Platform Deployment

(#include “Trusted Virtualization Platform”)

2) Attestation Implementation

2.1. Attestation to Virtual Machines

2.2. Attestation to Node Controller

KNOWN ISSUES:

A) **(SOLVED)** TPM not found in the specified-kernel bootstrapping mode of Qemu.
Description: The specified-kernel mode means that the kernel of the VM is specified as Qemu parameters (–kernel/-initrd/-appends) when bootstrapping instead of stored at the root image of the VM. When Qemu is bootstrapped with this mode, the TPM cannot be recognized inside the VM.
Solution: This is caused by a bug in trusted-seabios, the emulated BIOS for the VMs on Qemu. The parameters (–kernel/-initrd/-appends) broke the TPM related parts in the ACPI. Changes the size of TCPA(TPM) ACPI table in trusted-seabios from 64k to 48k solved this problem. (Solution provided by Stefan Berger, the author of the trusted-seabios).

B) **(SOLUTION PROPOSED)** Broken trusted chain in the specified-kernel bootstrapping mode.
Description: When the kernel is stored in the root image of a VM, it is bootstrapped by the bootloader (etc. Grub). The bootloader should be patched in this case with Trusted Computing supports to measure the kernel/initrd files when loading them. So then the trust chain can be extended to the kernel and to the applications hence force. However, in the specified-kernel mode, there is no dedicated bootloader. Kernel related files are loaded by Qemu before the vTPM is initialized by the seaBIOS. SeaBIOS then give controls to a simplified bootloader (linuxboot.bin), which does the major part of the kernel bootstrapping job, accessing the kernel directly to a specified memory address instead of loading them from files. Hence the kernels/initrd/cmdline-args are not measured in this case.

Proposed Solution: Modify the seaBIOS to first grab the memory addresses and sizes of the loaded kernel and related files from Qemu, and then calculate the measurement of these specified memory sections and extend them into the TPM. This action can be added after the last measurement operation done by seaBIOS (measures the option ROMS in this case), and before it gives control to linuxboot.bin.

C) **(SOLUTION PROPOSED) Disk image is not measured.**

Description: In a general platform (non-virtualization), the guarantee of the trusted binding from a hard disk to the OS is provided by the trusted BIOS, trusted bootloader, and the hardware's nature: BIOS measures ROMs which are bound to particular disks, and the bootloader resided on a particular disk loads a specified kernel. However, in the virtualized platform, the binding from the ROMs to the disk can be tampered. Moreover, no trusted bootloader exists in our scenario.

Proposed Solution 1: All the measurements to the VMs should be done after the vTPM has been initialised, which is done by the seaBIOS. The measurements to the root image should hence also be done within the seaBIOS. Codes can be added at the point of the disk-driver initialisation. In this case, parameters to Qemu specifying the files should be passed to seaBIOS. SeaBIOS also needs supports from Qemu to access the file in the physical hard disk for doing the measurement.

Proposed Solution 2: Measurements to the files (and possibly other parameters) are done by Qemu, and exended to the vTPM by seaBIOS.

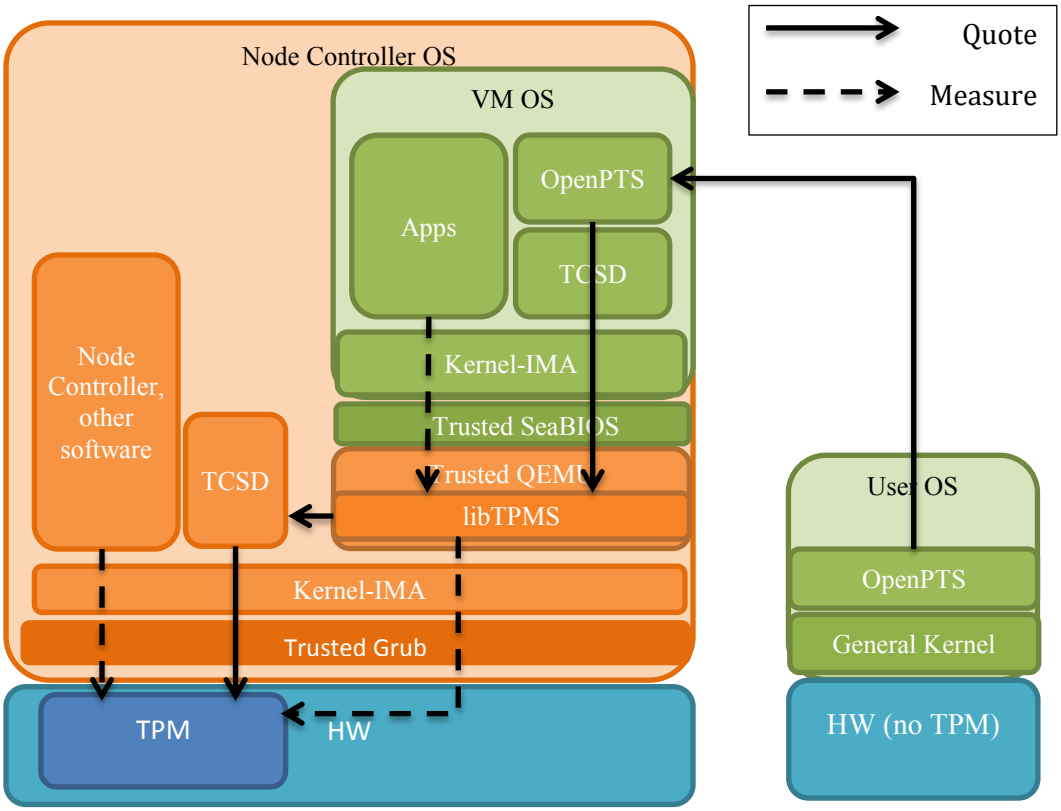

Figure 3. VM & Node Controller Attestation

2.3. Attestation to Storage Controller

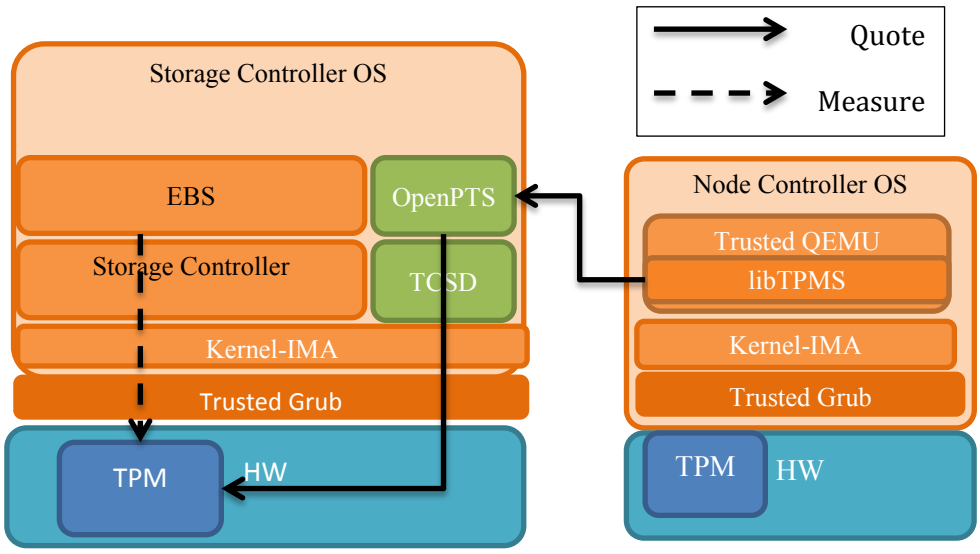

Figure 4. Storage Controller Attestation