Review of Trusted Cloud Computing Platform Security

S.R.Pojage
Dept of Computer Science and Engineering
PRMITR Amravati
India
sagarpojage@gmail.com

Dr.M.A.Pund
Dept of Computer Science and Engineering
PRMITR Amravati
India
mapund@mitra.ac.in

***Abstract* — Cloud computing infrastructures enable companies to cut costs by outsourcing computations on-demand. Security and privacy are two prime barriers to adoption of the cloud computing. Distributed Trusted Computing Platform (DTCP) model can improve the cloud computing security and will not bring much complexity to users. To address this problem on Infrastructure-as-a-Service model, a trusted cloud computing platform model has been proposed to provide a closed box execution environment that guarantees confidential execution of guest virtual machines. In this paper, we deal with different infrastructure level attacks and through the use of trusted cloud computing platform we provide a distributed solution to implement it.**

Keywords—cloud computing, cloud security, IaaSAttack, trusted cloud computing platform

1 INTRODUCTION

A Trusted Cloud computing

Cloud computing is an internet-based computing technology, where shared resources such as software, platform, storage and information are provided to customers on demand. Cloud computing is a computing platform for sharing resources that include infrastructures, software, applications, and business processes. Trusted computing is a technology developed and promoted by the Trusted Computing Group [7]. The term is taken from the field of trusted systems and has a specialized meaning. With Trusted Computing, the computer will consistently behave in expected ways, and those behaviors will be enforced by computer hardware and software. Enforcing this behavior is achieved by loading the hardware with a unique encryption key inaccessible to the rest of the system.

B Cloud Security

Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers. Organizations use the Cloud in a variety of different service models SaaS, PaaS, and IaaS and deployment models Private, Public, Hybrid, and Community).There are a number of security issues/concerns associated with cloud computing but these issues fall into two broad categories: security issues faced by cloud providers organizations providing software, platform or infrastructure-as-a-service via the cloud and security issues faced by their customers companies or organizations who host applications or store data on the cloud. The responsibility goes both ways, however: the provider must ensure that their infrastructure is secure and that their clients' data and applications are protected while the user must take measures to fortify their application and use strong passwords and authentication measures.

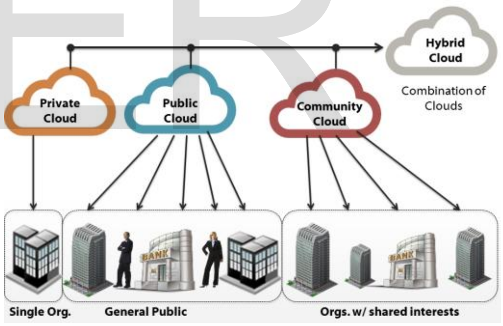

Figure Cloud deployment model

C Distributed Trusted Cloud Computing Platform

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 whether or not the IaaS enforces security. TCCP does the job of governing all trusted nodes on one entity only, instead the job is distributed among several entities, each managing a cluster, such that one single entity does become the failure of the complete system, and the system can not function smoothly. Distributed trusted cloud computing platform overcome TCCP problem.

2 .LITERATURE REVIEW

IT companies can greatly reduce costs of data management and data manipulation by loading such responsibilities on the shoulders of cloud computing services. Cloud computing features like agility, multi-tenancy, pervasiveness, elasticity and pay-per-use, make cloud computing an attractive platform to handle data of clients and relieve the client from such concerns. Enterprise level spending for on-premise solutions will descend, as cloud computing minimizes the need for licenses, software and hardware. In-spite of providing appreciable features, it has been found that clients sometimes feel reluctant to hand -over confidential data to the cloud providers. Data security is their major area of concern. Solidity and reliability are the characters a client is looking forward in a cloud provider. Hence ensuring data security is an indispensable feature a cloud provider must guarantee.

A Terra Architecture

It allows applications with a wide range of security requirements to run simultaneously on hardware. Applications on Terra enjoy the semantics of running on a separate, dedicated, tamper-resistant hardware platform, while retaining the ability to run with normal applications side-by-side on a general-purpose computing platform. These platforms can provide assurance of whether the VM is running on a single host, trusted by the third party. But many providers own data centers, where several machines are operating and a customer's VM can be dynamically assigned to any one of them. This complication and incomprehensibility resulting due to obscure backend of cloud service provider makes traditional platforms vulnerable to few dangers. Unfortunately, architecture like Terra has no answer to it.

To address these problems, some systems resort to specialized closed platforms, e.g. cellular phones, game consoles, and ATMs. Closed platforms give developers complete control over the structure and complexity of the software stack, thus they can tailor it to their security requirements. These platforms can provide hardware tamper resistance to ensure that the platform's software stack is not easily modified to make it misbehave. Embedded cryptographic keys permit these systems to identify their own software to remote systems, allowing them to make assumptions about the software's behavior. These capabilities allow closed platforms to offer higher assurance and address a wider range of threat models than current general-purpose platforms. The security benefits of starting from scratch on a "closed box" special-purpose platform can be significant. However, for most applications these benefits do not outweigh the advantages of general purpose open platforms that run many applications including a huge body of existing code and that take advantage of commodity hardware (CPU, storage, peripherals, etc.) that offers rich functionality and significant economies of scale. In this work, we describe a software architecture that attempts to resolve the conflict between these two approaches by supporting the capabilities of closed platforms on general-purpose computing hardware through a combination of hardware and operating system mechanisms.

In this architecture, called Terra, provides a simple and flexible programming model that allows application designers to build secure applications in the same way they would on a dedicated closed platform. At the same time, Terra supports today's operating systems and applications. Terra realizes this union with a *trusted virtual machine monitor* (TVMM), that is, a high-assurance virtual machine monitor that partitions a single tamper-resistant, general-purpose platform into multiple isolated virtual machines. Using a TVMM, existing applications and operating systems can each run in a standard virtual machine ("open-box VM") that provides the semantics of today's open platforms. Applications can also run in their own closed-box virtual machines ("closed-box VMs") that provide the functionality of running on a dedicated closed platform. The TVMM protects the privacy and integrity of a closed-box VM's contents. Applications running inside a closed-box VM can tailor their software stacks to their security requirements. Finally, the TVMM allows applications to cryptographically authenticate the running software stack to remote parties in a process called attestation.Both open- and closed-box VMs provide a raw hardware interface that is practically identical to the underlying physical machine. Thus, VMs can run all existing commodity software that would normally run on the hardware. Because a hardware-level interface is provided, application designers can completely specify what software runs inside a VM, allowing them to tailor an application's software stack to its security, compatibility, and performance needs. Closed-box VMs are isolated from the rest of the platform. Through hardware memory protection and cryptographic protection of storage, their contents are protected from observation and tampering by the platform owner and malicious parties.

At the heart of Terra is a virtual machine monitor (VMM). Like any VMM, Terra virtualizes machine resources to allow many virtual machines (VMs) to run independently and concurrently. Terra also provides additional security capabilities including acting as a trusted party to authenticate the software running in a VM to remote parties. Because of this property we refer to it as a "trusted VMM" (TVMM). At a high level, the TVMM exports two VM abstractions. Open box VMs provide the semantics of today's open platforms. These can run commodity operating systems and provide the appearance of today's general-purpose platforms. Closed-box VMs implement the semantics of a closed-box platform. Their content cannot be inspected or manipulated by the platform owner. Thus, their content is secure, neither inspectable nor modifiable by any but those who constructed it, who can explicitly provide themselves access.

Terra provides a raw virtual machine as the development target for applications, lending great flexibility to application designers. Applications can be designed from the (virtual) hardware up, using the operating systems that best suit their security, portability, and efficiency needs. Operating systems that run in VMs may be as simple as a bootstrap loader plus application code or as complex as a commodity operating system that runs only one application. Applications can completely tailor the OS to their security needs. Instead of running single closed-box applications, a closed-box VM might run a special trusted OS with a selection of applications designed specifically for it, thus providing something similar to the NGSCB model. VMs on a single physical machine communicate with one another over virtualized standard I/O interfaces such as NICs, serial ports, etc. The VMM can also multiplex the display and input devices. Thus, from the user's perspective, a closed-box VM may take on the appearance of a normal application, a virtual network appliance, or a virtual device (e.g. a USB device). The responsibility for configuring how these VMs are granted storage and memory, connected, started, stopped, etc. is delegated to a special management *VM*. The TVMM offers the management VM a basic interface to carry out these tasks. Where the TVMM provides resource management mechanisms, the management VM decides policy, providing a higher-level interface to users and other VMs.

B Trusted Platform Module (TPM)

Hardware virtualization has enjoyed a rapid resurgence in recent years as a way to reduce the total cost of ownership of computer systems [5]. This resurgence is specially apparent in corporate data centers such as web hosting centers, where sharing each hardware platform among multiple software workloads leads to improved utilization and reduced operating expenses. However, along with these cost benefits come added security concerns. Workloads that share the same platform
must often be kept separate for a multitude of reasons. For example, government regulations may require an investment bank to maintain a strict separation between its market analysis and security underwriting departments, including their respective information processing facilities. Similarly, commercial interests may dictate that the web sites of competing businesses not have access to each other's data. In addition, concerns about malicious software subverting normal operations become specially acute in these shared hardware environments. For example, a remote client of a medical services site would like to determine that the server is not running corrupted software that will expose private information to a third party or return wrong medical information. The increasing use of virtualization thus gives rise to stringent security requirements in the areas of software integrity and workload isolation.

The combination of a hardware-based root of trust such as the Trusted Platform Module (TPM), and a virtual machine-based system such as Xen VMware, or PHYP, is exceedingly well suited to satisfying these security requirements. Virtual machine monitors, or hypervisors, are naturally good at isolating workloads from each other because they mediate all access to physical resources by virtual machines. A hardware root of trust is resistant to software attacks and provides a basis for reasoning about the integrity of all software running on a platform, from the hypervisor itself to all operating systems and applications running inside virtual machines. In particular, the TPM enables remote attestation by digitally signing cryptographic hashes of software components. In this context, attestation means to affirm that some software or hardware is genuine or correct. TPM chips are widely deployed on laptop and desktop PCs, and are becoming increasingly available on server-class machines such as the IBM eServer x366 . Virtualizing the TPM is necessary to make its capabilities available to all virtual machines running on a platform.

Each virtual machine with need of TPM functionality should be made to feel that it has access to its own private TPM, even though there may be many more virtual machines than physical TPMs on the system (typically there is a single hardware TPM per platform). It is thus necessary to create multiple virtual TPM instances, each of which faithfully emulates the functions of hardware TPM. However, virtualizing the TPM presents difficult challenges because of the need to preserve its security properties. The difficulty lies not in providing the low-level TPM command set, but in properly supporting higher level security concepts such as trust establishment. In particular, it is necessary to extend the chain of trust from the physical TPM to each virtual TPM via careful management of signing keys and certificates. As a result, some application and operating system software that relies on TPM functionality needs to be made aware of semantic differences between virtual and physical TPMs, so that certificate chains can be correctly built and evaluated, and trust chains correctly established and followed. An additional challenge is the need to support migration of a virtual TPM instance between hardware platforms when its associated virtual machine migrates. The ability to suspend, migrate, and resume virtual machines is an important benefit of hardware virtualization. For the virtual TPM, migration requires protecting the secrecy and integrity of data stored in a virtual TPM instance during the transfer between platforms, and re-establishing the chain of trust on the new platform.

The TPM is a security specification defined by the Trusted Computing Group. Its implementation is available as a chip that is physically attached to a platform's motherboard and controlled by software running on the system using well-defined commands. It provides cryptographic operations such as asymmetric key

generation, decryption, encryption, signing and migration of keys between TPMs, as well as random number generation and hashing. It also provides secure storage for small amounts of information such as cryptographic keys. Because the TPM is implemented in hardware and presents a carefully designed interface, it is resistant to software attacks . Of particular interest is the Platform Configuration Register (PCR) *extension* operation. PCRs are initialized at power up and can only be modified by reset or extension. The PCR extension function cryptographically updates a PCR using the following function:
Extend (PCRN, value) = SHA1 (PCRN||value)

The cryptographic properties of the extension operation state that it is infeasible to reach a certain PCR state through two different sequences of values. SHA1 refers to the Secure Hash Algorithm standard [19]. The || operation represents a concatenation of two byte arrays. PCR extensions are used during the platform boot process and start within early-executed code in the Basic Input/ Output System (BIOS) that is referred to as the Core Root of Trust for Measurement (CRTM).

Hash values of byte arrays representing code or configuration data are calculated, or *measured*, and PCRs are extended with these values. A final PCR value represents this accumulation of a unique sequence of measurements. Along with a sequential list of individual measurements and applications' names and information about measured configuration data, PCR values are used to decide whether a system can be trusted. A transitive trust model is implemented that hands off the measuring from the BIOS to the boot loader and finally to the operating system. Procedures have also been developed for operating systems to measure launched applications, scripts and configuration files.

Besides the aforementioned cryptographic operations it is possible to *seal* information against the state of the TPM, where its state is represented through a subset of PCRs. Sealed information is encrypted with a public key and can only be decrypted if the selected PCRs are in the exact state that they were at the time of sealing. There are a number of signing keys associated with a TPM. Each TPM can be identified by a unique built-in key, the Endorsement Key (EK), which stands for the validity of the TPM [7]. The device manufacturer should provide a certificate for the EK. Related to the EK are Attestation Identity Keys (AIKs). An AIK is created by the TPM and linked to the local platform through a certificate for that AIK. This certificate is created and signed by a certificate authority (CA). In particular, a *privacy CA* allows a platform to present different AIKs to different remote parties, so that it is impossible for these parties to determine that the AIKs are coming from the same platform. AIKs are primarily used during quote operations to provide a signature over a subset of PCRs as well as a 160-bit nonce. Quotes are delivered to remote parties to enable them to verify properties of the platform.

VMMs [8], also known as hypervisors, allow multiple operating systems to simultaneously run on one machine.
A VMM is a software layer underneath the operating system that meets two basic requirements:
• It provides a Virtual Machine (VM) abstraction that models and emulates a physical machine.
• It provides isolation between virtual machines. The basic responsibility of a VMM is to provide CPU time, memory and interrupts to each VM.
It needs to set up the page tables and memory management unit of the CPU such that each VM runs in its own isolated sandbox The hypervisor itself remains in full control over the resources given to a VM. During the boot process of a VMM, often an initial virtual machine is started that serves as a management system for starting further virtual machines. Depending on the fidelity of the emulation of a physical machine, it may be necessary to make modifications to an operating system for it to run on a VMM. If modifications are required the environment is said to be *paravirtualized*, otherwise the VMM is said to provide a *fully virtualized* environment.

C Trusted Cloud Computing Platform (TCCP)

Companies can greatly reduce IT costs by offloading data and computation to cloud computing services. Still, many companies are reluctant to do so, mostly due to outstanding security concerns. One of the most serious concerns is the possibility of confidentiality violations. Either maliciously or accidentally, cloud provider's employees can tamper with or leak a company's data. Such actions can severely damage the reputation or finances of a company. In order to prevent confidentiality violations, cloud services' customers might resort 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, the provider hosts virtual machines (VMs) on behalf of its customers, who can do arbitrary computations. In these systems, anyone with privileged access to the host can read or manipulate a customer's data. Consequently, customers cannot protect their VMs on their own. Cloud service providers are making a substantial effort to secure their systems, in order to minimize the threat of insider attacks, and reinforce the confidence of customers. For example, they protect and restrict access to the hardware facilities, adopt stringent accountability and auditing procedures, and minimize the number of staff who has access to critical components of the infrastructure [8]. 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 guarantees the confidentiality and integrity of computation, in a way that is verifiable by the customers of the service Traditional trusted computing platforms like Terra [4] take a compelling approach to this problem. For example, Terra is able to prevent the owner of a physical host from inspecting and interfering with a computation. Terra also provides a remote attestation capability that enables a remote party to determine upfront whether the host can securely run the computation. This mechanism reliably detects whether or not the host is running a platform implementation that the remote party trusts. These platforms can effectively secure a VM running in a single host. However, many providers run data centers comprising several hundreds of machines, and a customer's VM can be dynamically scheduled to run on any one of them. This complexity and the opaqueness of the provider backend create vulnerabilities that traditional trusted platforms cannot address. 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.

Infrastructure as a Service

Today, myriads of cloud providers offer services at various layers of the software stack. At lower layers, Infrasructure 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 in guaranteeing the confidentiality of computations increases for services sitting on higher layers of the software stack, because services themselves provide and run the software that directly manipulates customer's data (e.g., Google Docs). We focus on the lower layer IaaS cloud providers where securing a customer's VM is more manageable.While very little detail is known about the internal organization of commercial IaaS services, we escribe (andbase our proposal on) Eucalyptus [6], an open source IaaS platform that offers an interface similar to EC2 This system manages one or more clusters whose nodes run a virtual machine monitor (typically Xen) to host customers' VMs. Eucalyptus comprehends a set of components to manage the clusters. For simplicity, our description aggregates all these components in a single cloud manager (CM) that handles a single cluster From the perspective of users, Eucalyptus provides a web service interface to launch, manage, and terminate VMs. A VM is launched from a virtual machine image (VMI) loaded from the CM. Once a VM is launched, users can log in to it using normal tools such asssh. Aside from the interface to every user, the CM exports services that can be used to perform administrative tasks such as adding and removing VMIs or users. Xen supports live migration, allowing a VM to shift its physical host while still running, in a way that is transparent to the user. Migration can be useful for resource consolidation or load balancing within the cluster.

Attack model

A sysadmin of the cloud provider that has privileged control over the backend can perpetrate many attacks in order to access the memory of a customer's VM. With root privileges at each machine, the sysadmin can install or execute all sorts of software to perform an attack. For example, if Xen is used at the backend, Xenaccess [7] allows a sysadmin to run a user level process in Dom0 that directly accesses the content of a VM's memory at run time. Furthermore, with physical access to the machine, a sysadmin can perform more sophisticated attacks like cold boot attacks and even tamper with the hardware. In current IaaS providers, we can reasonably consider that no single person accumulates all these privileges. Moreover, providers already deploy stringent security devices, restricted access control policies, and surveillance mechanisms to protect the physical integrity of the hardware. Thus, we assume that, by enforcing a security perimeter, the provider itself can prevent attacks that require physical access to the machines. Nevertheless, sysadmins need privileged permissions at the cluster's machines in order to manage the software they run. Since we do not precisely know the praxis of current IaaS providers, we assume in our 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 privi- leges remotely logged to a machine hosting a VM cannot access its memory

Trusted Computing

The Trusted Computing Group (TCG) [7] 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 contains an endorsement private key (EK) that uniquely identifies the TPM (thus, the physical host), and some cryptographic functions that cannot be modified. The respective manufacturers sign the corresponding public key to guarantee the correctness of the chip and validity of the key. Trusted platforms [1, 4, 5, 9] leverage the features of TPM chips to enable *remote attestation*. This mechanism works as follows. At boot time, the host computes a measurement listML 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 attest to the platform, a remote party challenges the platform running at the host with a nonce nU. The platform asks the local TPM 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

3. CONCLUSION

To improve trusted cloud computing privacy, security, sensitivity and storage efficiency of data and computation are major obstacles for organization willing to adopt the services of cloud computing. The design of Distributed Trusted Cloud Computing Platform (DTCCP) which acts as solution to suggest the cloud user that the platform on which they wish to run their computations is indeed trusted or not. It allows IaaS services such as Amazon EC2 to provide an enclosed execution environment for its users. DTCCP grants private execution environment to the guest VMs, and enables the users to attest to the IaaS provider well in advance, if at all the provider can provide a secure platform for their VM execution. Also DTCCP model does not implement the job of governing all trusted nodes on one entity only, instead the job is distributed among several entities, each managing a cluster, such that one single entity become fail still entire system carried out work smoothly.

REFERENCES:
[1] N. Santos, K.P. Gummadi, R. Rodrigues, "Towards Trusted Cloud Computing". In Proc. of the 1s USENIX Workshop on Hot Topics in Cloud Computing, Berkeley, CA, USA, 2009.

[2] Louis Columbus, "Roundup of Cloud Computing Forecasts and Market Estimates, 2012", January 17, 2012. [Online]. Available: http://softwarestrategiesblog.com/2012/01/17/roundup-of-cloud-computing-forecasts-and-market-estimates-2012/

[3] B.D Payne,M.Carbone ,and W Lee ,"Secure and Flexible Monitoring of Virtual Machines, " In Proc. Of ACSAC07,2007

[4] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, D. Boneh,"Terra: A Virtual Machine-Based Platform for Trusted Computing". In Pro c. of SOSP'03, 2003.

[5] Hamid Banirostam, Alireza Hedayati, Ahmad Khadem Zadeh, Elham Shamsinezha d, "A Trust Based Approach for Increasing Security in Cloud Computing Infrastructure". In UKSim 15th International Conference on Computer Modelling and Simulation, 2013.

[6] F. John Krautheim, Dhan anjay S. Phatak, and Alan T. Sherman, "Introducing the Trusted Virtual Environment Module: A New Mechanism for Rooting Trust in Cloud Computing". In TRUST 2010, LNCS 6101, pp. 211–227, 2010, © Springer-Verlag B erlin Heidelberg 2010.

[7] TCG. [Online]. Available: http://www.trustedcomputinggroup.org/

[8] Stefan Berger, Ram´on C ´aceres, Kenneth A. Goldman, Ronald Perez, Reiner Saile r, Leendert van Doorn, "vTPM: Virtualizing the Trusted Platform Module". In Proc. of USENIX-SS'06, Berkeley, CA, USA, 2006.

[9] Partha Sen, Pritam Saha, Sunirmal Khatua "A Distributed Approach towards Trusted Cloud Computing platform" in 2015 Applications and Innovations in Mobile Computing (AIMoC)