Introducing the Trusted Virtual Environment Module: A New Mechanism for Rooting Trust in Cloud Computing

F. John Krautheim[*], Dhananjay S. Phatak, and Alan T. Sherman[*]

Cyber Defense Lab, Dept. of CSEE
University of Maryland, Baltimore County (UMBC)
Baltimore, MD, USA 21250
{john.krautheim,phatak,sherman}@umbc.edu

Abstract. We introduce a new mechanism for rooting trust in a cloud computing environment called the *Trusted Virtual Environment Module (TVEM)*. The TVEM helps solve the core security challenge of cloud computing by enabling parties to establish trust relationships where an information owner creates and runs a virtual environment on a platform owned by a separate service provider. The TVEM is a software appliance that provides enhanced features for cloud virtual environments over existing Trusted Platform Module virtualization techniques, which includes an improved application program interface, cryptographic algorithm flexibility, and a configurable modular architecture. We define a unique Trusted Environment Key that combines trust from the information owner and the service provider to create a dual root of trust for the TVEM that is distinct for every virtual environment and separate from the platform's trust. This paper presents the requirements, design, and architecture of our approach.

Keywords: cloud computing, trust, security, virtualization, TPM.

1 Introduction

Cloud computing is changing the landscape of corporate computing. As companies turn to cloud services to reduce costs compared to their internally managed *Information Technology (IT)* systems, a fundamental shift is occurring in the way IT and computing services are delivered and purchased [1]. With this shift towards *utility computing* [2], new trust relationships arise that force the parties to reconsider the way we handle and manage information in the cloud.

Krautheim, *et al.* [3] define the *Private Virtual Infrastructure (PVI)* cloud trust model describing the unique trust relationships that occur in *Infrastructure as a Service (IaaS)* [4] cloud computing environments. This paper applies the PVI cloud trust model to IaaS clouds with our new *Trusted Virtual Environment Module (TVEM)* and *Virtual Trust Network (VTN)*.

[*] Krautheim and Sherman are supported in part by the Department of Defense under grants H98230-08-1-0334 and H98230-09-1-0404.

In IaaS cloud computing, an *information owner*, or client, rents virtual computing resources in the form of a *Virtual Machine (VM)* on a host platform operated by a second party *service provider*. The information owner wishes to protect private and sensitive data that are processed in the virtual environment on the rented VM. The *virtual environment* is the entity that is controlled by the information owner and consists of *all* software components, from the *Operating System (OS)* to the applications, that execute on the VM. To assure the information is protected, the client needs to verify the trustworthiness of the host platform and virtual environment. The TVEM and VTN provide the mechanisms to verify the host platform and virtual environment within an IaaS cloud and report the results back to an information owner. No current capability exists to perform these functions.

A current means for establishing trust in computing platforms is the *Trusted Platform Module (TPM)*, a core component of the *root of trust* for the platform. A root of trust is a component of a computing platform that is implicitly trusted to provide a specified set of controlled functions to measure and pass control to other platform components [5]. TPMs are designed to support a single OS on a single platform and typically do not scale well when virtualization is introduced to the platform [6]. Support for multiple virtual environments that simultaneously access TPM resources is required. A *Virtual TPM (VTPM)* that replicates the physical resources of a TPM in software is one method of virtualizing the TPM functions for sharing among multiple virtual environments.

Krautheim's *Locater Bot (LoBot)* [3, 7] uses the VTPM to root trust for a virtual environment in a PVI; however, the VTPM implementation has several issues that make it problematic to use as a root of trust for cloud virtual environments. Three major shortcomings of the VTPM are: the VTPM's trust is rooted to the physical platform on which it is operating, which is typically not owned by the information owner; a VTPM must follow the TPM specification [8], which includes extraneous functionality that is not useful for virtual environments; and a VTPM has non-persistent storage, meaning that it loses all keys, settings, and non-volatile storage upon termination. The TVEM solves these problems through application of the PVI cloud trust model, providing a modular and extensible architecture that allows algorithm and function flexibility, and providing persistent storage for keys, non-volatile memory and settings.

The core challenge in cloud computing that TVEM solves is establishing trust that is distinct for the virtual environment and separate from the hosting platform. *Virtual environment trust* is defined as trust in the virtual environment that is a combination of trust in the service provider's platform and trust from the information owner's domain. Virtual environment trust is necessary to convey ownership and protect information in the cloud. To implement this virtual environment trust, a *Trusted Environment Key (TEK)* is defined and used as the *Endorsement Key (EK)* for the TVEM. The TEK, like the EK, is a unique value used as the *Root of Trust for Reporting (RTR)* to identify the TVEM and attest the virtual environment. The TEK is generated by the virtual environment owner and secured with the service provider's platform storage key creating a *compound trust* distinct and separate from the platform.

The TVEM is a software appliance that is implemented as a helper, or stub, VM. The TVEM is protected by hardware enforced memory and process isolation via

Intel's *Virtualization Technology for Directed I/O (VT-d)* [9] and *Trusted eXecution Technology (TXT)* [10]. The TVEM provides attestation support and trusted storage for the virtual environment similar to functionality provided by VTPM; however, the TVEM does not have to conform to the TPM specification enabling the TVEM to be extensible through functional and cryptographic algorithm flexibility in a configurable modular architecture. The TVEM has multiple interfaces, including an *Application Program Interface (API)*, which moves the *Trusted Software Stack (TSS)* into the TVEM eliminating the burden on the virtual environment to implement the TSS. The API provides for hardened and lightweight environments and reduces the opportunities of implementation errors. These capabilities allow system designers to customize the TVEM and virtual environment to meet their information confidentiality and integrity requirements.

TVEMs are not stand alone devices; they are part of a system to implement trust in cloud computing. The system includes: the TVEM; a TVEM manager in the host hypervisor for host platform TPM access and TVEM provisioning; a VTN control plane that provides system management and support for persistent storage; and a *TVEM Factory (TF)* to manufacture TVEMs, manage keys, and provision TVEMs securely on host platforms.

2 Motivation

Utility cloud computing can provide many benefits to companies wishing to reduce their IT expenses and overhead. Security of information in the cloud and the trustworthiness of the cloud environment is a major concern with IaaS clouds. We describe an example IaaS cloud computing application: a cloud web server. This application benefits from using the TVEM and VTN to manage trust.

A virtual web server in the cloud has many benefits over maintaining a web server locally, a significant advantage being increased availability. The cloud's always-on presence and location flexibility enhance the availability of a web server by providing scalability, migratability, and redundancy. If a server is overwhelmed with requests, it can be migrated to a platform that has increased capacity, or new instances of the server can be instantiated to handle the increased load. Migration and failure restart can be used if host hardware fails or Internet service becomes unavailable.

A server certificate is a critical piece of data on the web server that authenticates its owner. If a company wants to prove that it is the owner of web server, it would obtain an *Extended Validation (EV)* certificate and a *Secure Socket Layer (SSL)* certificate from a certificate authority. The EV and SSL certificates have a public and private key portion that a guest may use to verify the server owner and establish an encrypted SSL session with a server. In a cloud environment, the identity of the web server owner and the service provider needs to be differentiated, which is accomplished via the certificates. The certificates should be accessible only by the web server owner and must be protected from the service provider and other users of the cloud service. If the private portions of the keys are disclosed, anyone who gains access to the private keys can purport to be a valid web server for the information owner. If the private key is stored on a public cloud service, anyone with access to the system could possibly access the key; therefore, the owner of the certificate needs to keep the private key protected from compromise by the service provider, other cloud users, and attackers on the Internet. TVEM protects SSL certificates on a cloud web server by encrypting the certificate such that is accessible only by the TVEM and decrypted inside the host platform's TPM ensuring the plaintext key cannot be observed.

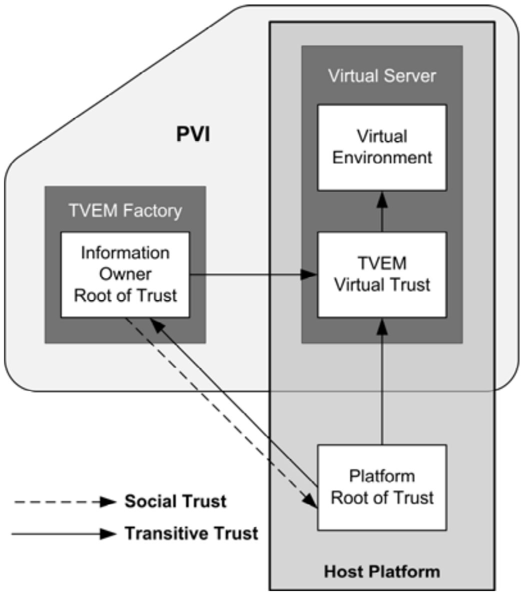

Fig. 1. Trust relationships in a cloud computing environment consist of inferred trust (social trust) and inherited trust (technical trust). The TVEM's virtual trust is a combination of the information owner's trust and host platform trust.

3 Trust in the Cloud

Trust in cloud computing is more complex than in a traditional IT scenario where the information owner owns his own computers. Fig. 1 shows the trust relationships in an IasS cloud as defined by the PVI cloud trust model. The trust chain combines trust from the information owner's domain (or PVI) and trust from the service provider's platform into the virtual environment trust. The information owner has an inferred trust in the platform from a social trust relationship with the service providers. The information owner root of trust is established in the TF and is the core root of trust for the entire PVI. The TF needs to inherit trust from the host platform root of trust measurements to ensure that the PVI is being implemented on a trustworthy platform. The PVI combines the inherited platform trust and information owner trust in a TEK and places it in the TVEM. The TVEM trust is provisioned on the host platform such that it is bound to the platform root of trust, creating the dual rooted virtual environment trust. If either root of trust is revoked, the virtual environment trust is invalidated.

3.1 Social Trust

Social trust is a trust that arises between two entities based upon social relationships. Social trust is established based upon reputations, previous interactions, and contractual obligations. There are two critical social trust relationships that must be established in cloud computing from the perspective of the information owner: service provider trust and cloud user trust. Social trust cannot be measured, but is important to build confidence that an entity is holding up its end of a contract.

Service provider trust lies in the relationship between customer and vendor. If the provider has a good reputation, then there is sufficient reason for customers to trust the provider. A vendor that has questionable service or ethics would not be as trustworthy as a vendor with excellent service and ethics.

Cloud user trust is the amount of trust the user places in the services delivered via the cloud. The user has to be confident that the system is going to protect their data, transactions, and privacy. The user's trust is a social trust in the information owner. The information owner must assure that the services being provided meet the user's expectations.

3.2 Technical Trust

In a cloud computing environment, multiple entities must trust the cloud services: the user of the cloud service or information owner, the provider of the cloud service, and third parties. PVI defined a new paradigm of cloud computing that separates the security responsibility between the service provider and information owner and accounted for third parties. A third party is an outside entity that is providing service to or receiving services from either the user or service provider.

The cloud trust model is based on *transitive trust*, which is the notion that if entity A trusts entity B and entity B trusts entity C, then entity A trusts entity C. This property allows a chain of trust to be built from a single root of trust.

Information owner trust is the foundation of trust that the information owner places in the PVI. Information owner trust is implemented by the TF. Since the information owner has physical control of the TF, the configuration of the TF is a known quantity and can be used as the root of trust for the PVI. As long as the information owner maintains trust in the TF, trust can be established in the PVI and used to build trust chains with cloud host platforms.

Host platform trust lies in the hardware trust of the platform and is measurable. Trustworthiness starts with *Core Root of Trust for Measurement (CRTM)* of the platform. The CRTM is the core set of instructions run at boot or reset that are responsible for establishing trust in the system by measuring the BIOS, and then passing control to the measured BIOS and rest of the *Trusted Computing Base (TCB)* of the platform. The TCB consists of all measured components that provide the foundation of trust in the platform. Platform trustworthiness is determined by an outside entity via attestation from the TPM, which is the *Root of Trust for Reporting (RTR)* on the hosting platform. The TPM also serves as the *Root of Trust of Storage (RTS)* of the platform that is implicitly trusted to store information securely. The attestation from the TPM provides evidence of the state of the platform, from which other entities can decide whether to trust the platform.

Cloud virtual environment trust is the amount of trust placed in the virtual environments created in the cloud. Virtual environment trust is measureable, but there are complications in a cloud environment where the information owner's requirements are different than the platform owner's.

4 Related Work

There are many issues that must be solved to virtualize a TPM. The limited resources of the TPM must be either shared or replicated for each virtualized TPM. Specifically, resources that cannot be shared on the TPM are the EK, *Platform Configuration Registers (PCRs)*, and non-volatile storage. These resources must be replicated by every VTPM implementation.

A common approach to virtualizing the TPM has been to emulate the TPM in software and provide an instance for each virtual environment. The VTPM can be bound to a physical TPM for additional security. Berger, *et al.,* [11] took this approach for their vTPM implementation along with an additional approach of using an IBM 4758 Cryptographic Coprocessor to implement the vTPMs. Scarlata [6] followed with a framework for TPM virtualization, which described a VTPM framework for emulating TPMs in software.

England [12] took a different approach to TPM virtualization with paravirtualized TPM sharing. Paravirtualization is a technique used by the Xen hypervisor [13] to present a software interface to the VM that is similar to the underlying hardware and requires the OS to be modified. This method uses the hypervisor to mediate access to a single hardware TPM. The hypervisor shadows the PCRs for each virtual environment thus overcoming the PCR limitation. This design reduces the ability for migration since the virtualization is done in the hypervisor and uses physical TPM resources that are not transferrable to other platforms.

Another unique approach is property-based TPM virtualization by Sadeghi [14]. This technique uses a different methodology to measure the platform's state and generate keys. Properties are measured for reporting the state of the platform, which are less susceptible to changes in software configuration updates and patches, and makes migration easier.

The Berlios TPM emulator [15] is a form of TPM virtualization, providing a software emulation of a hardware TPM. The TPM emulator can provide TPM services to virtual environments, but does not have any binding to the hardware, limiting its ability for operational use. Consequently, the Berlios TPM emulator is useful for development purposes only.

5 Design Considerations

We explain our design considerations for the TVEM in this section. We considered multiple approaches to implementing a trust module for virtual environments and realized the best way to ensure that each virtual environment has a trust module is for the module to be implemented in software.

By implementing the TVEM in software, we do not have cost, physical, or resource restrictions. The TVEM design is bound by memory and computation restrictions, which are much less restrictive than physical restrictions. On a typical host platform, we can provide multiple fully functional, uncompromised TVEMs for many virtual machines at the same time.

There are several advantages to implementing a trust module in software versus hardware. A software platform can be changed to accommodate vulnerabilities that are discovered after release (*e.g.*, SHA1 collision attacks [16]). A software module can also support different algorithms for different applications and locations, which is important in cloud environments. Export controls on cryptographic algorithms may dictate that a certain algorithm may not be used in certain countries. With the worldwide presence of the cloud, algorithm flexibility is essential. A cloud environment in a restricted country will need an algorithm allowed to be exported, while a stronger algorithm may be used on a system in another country where more security is permitted. An advantage to a modular software design of the TVEM is flexibility to use algorithms that are compatible with the current TPM specification or use new algorithms for future applications and enhanced security. This flexibility allows the TVEM to be used in applications where features of TPM.Next are required, but the hardware does not support TPM.Next.

5.1 Threat Model

The TVEM must protect data and operations from attack. Since the TVEM will be implemented in software, there are multiple attack vectors. Potentially any code running on the host platform could attack the TVEM. We assume VT-d and TXT hardware isolation are in place, which protects the module from attack by entities with access to the platform. The entities that have access to the platform include the following: the service provider, who has root access to the platform; other cloud users on the same system or within the same cloud environment; and outside attackers that access the system via the Internet.

A malicious program may gain access to private data, including keys, inside of the TVEM. The malicious program can then modify and substitute data, to include replacing keys, modifying hashes, and state information. The code of the TVEM could be modified by replacing strong cryptographic algorithms with weaker ones. An attacker could also reduce the entropy of the *random number generator (RNG)* thus reducing the effectiveness of the keys. The state could also be modified to a known or predetermined state to weaken cryptographic results.

The TVEM's main defense against attacks is robust isolation and state verification. Features of the trusted platform can be used to protect against many attacks including: software modification, malicious code, key exposure and modification, and VM isolation and containment attacks, without detection.

The TVEM cannot defend against hardware attacks since it is a software module. A sophisticated attacker with control over hardware would be able to circumvent the TVEMs security, gain access to protected information in the TVEM, and modify state and/or instructions inside the TVEM to alter outcomes of the TVEM's operations.

5.2 Deployment Model

The deployment model for the TVEM is to build and maintain TVEMs on the TF in a secure location that is under the information owner's full physical control. The TF should be isolated and physically separated from the service provider's facility to ensure that it cannot be compromised.

Fully self contained TVEMs images are configured and built in the TF. The TVEM image is provisioned on the service provider's platform through the secure provisioning protocol described in [3] that ensures the TVEM is loaded on the correct host without modification and launched securely. The protocol ensures that the code launched was exactly the code configured by the TF so that the TVEM will operate as intended by its owner.

The service provider needs to accommodate the deployment of TVEMs by providing a TVEM Manager that is accessible by the customer for interfacing with the host TPM. The TPM access can be provided by delegating the customer certain privileged operations to configure the TVEM and interface with the TPM.

6 TVEM System Architecture

The TVEM is a software trust module for providing trust services to a VM or virtual environment in an IaaS cloud computing environment. The TVEM is the protection module and root of trust for a virtual environment that is in a remote location and the virtual environment has the ability to migrate to other platforms; therefore, it is not possible to implement a TVEM in hardware. Thus, a software implementation is the only solution for the TVEM.

Because the TVEM is a cryptographic module and data confidentiality is of utmost importance; we chose for the TVEM to be a self-contained virtual appliance that is implemented in a helper VM or stub domain.

Fig. 2 shows the configuration of a *Host Platform (HP)* with multiple virtual environments that require a TVEM. The virtual environment may be an entire virtualized OS that supports many applications or a special purpose virtual environment that performs a single application. The TVEM lies between the hypervisor and its associated VM. The hypervisor must be aware of TVEMs and provide support via a TVEM manager. The TVEM manager provides mediation for TPM services from each TVEM and other processes that require TPM services. The host platform must provide the TVEM manager and allow access for TVEMs.

The host platform's TPM is used as the RTS to secure the TVEM's private information on the HP. A transitive trust chain is built from the TPM through the hypervisor and TVEM manager to the TVEM ensuring trust in the TVEM is rooted in the hardware trust of the platform.

To ensure that data within the TVEM remains private, several security mechanisms need to be in place on the host platform. An isolating hypervisor such as sHype [17] can be used along with Intel VT-d and TXT to provide additional protection through hardware isolation.

6.1 TVEM Manager

The TVEM manager is the hub for the VTN on the host platform. The TVEM manager is responsible for routing VTN communications between the TF and VTEMs and for arbitrating all access between TVEMs on the platform and the host TPM. The TF communicates with host platforms through the TVEM Managers.

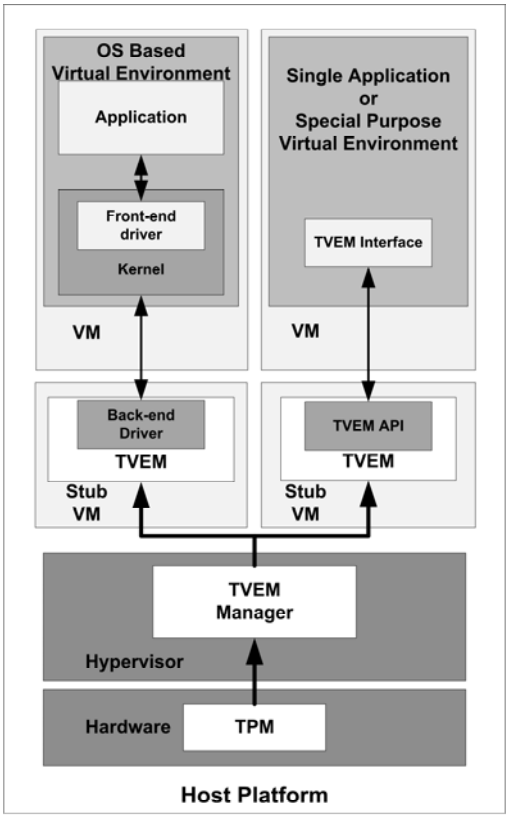

Fig. 2. The host platform has a single TPM, hypervisor, and TVEM manager supporting multiple virtual environments each in its own tightly coupled TVEM

Each host must have a TVEM manager that provides an interface to the host TPM. The TVEM manager must be placed in the hypervisor on the host platform so that it may have access to the host TPM and provide provisioning functions required to support TVEMs. Host TPM access is required for reading the platform PCRs and SRK so that TVEMs may be bound to the host TPM.

Importantly, the TVEM manager is part of the host platform, it is owned by the service provider and is not part of the information owner's domain (see Fig. 2); therefore, the TVEM manager must be a trusted component and part of the measured configuration on the host platform.

A TVEM manager on a host platform may support multiple VTNs from the same information owner or VTNs from other information owners simultaneously. The TVEM manager must be able to isolate communication from multiple VTNs and allow access only to TVEMs associated with the proper VTN.

6.2 Trusted Environment Key

The *Trusted Environment Key (TEK)* is critical in providing security and trust for the TVEM. It prevents cloning of the TVEM and protects the contents of the TVEM from the platform owner and other processes on the platform. The TEK is a unique key generated for a TVEM. The TEK is the TVEM's endorsement key and serves the same purpose as a TPM's EK. The TEK is generated from the VTN root certificate in the *TVEM Factory (TF).*

The TF generates a VTN certificate, which is the parent for the all TVEM TEKs in a VTN. The VTN certificate is defined as:

$$VTN_{TF} = SRK_{TF}\{VTN\}$$

The TF's TPM generates a unique VTN certificate for each VTN protecting it with the TPM's SRK. The VTN key is a *Migratable Storage Key (MSK)* that can be transferred to other TPMs. The TEK is then generated as a child of the VTN key and is thus a MSK as well. Both keys are transferred to the *Host Platform's (HP)* through the key migration process. The TEK is defined as:

$$TEK_{HP} = MSK_{HP}\{VTN_{TF}\{TEK\}\}$$

The TF migrates the TEK and VTN key to the HP binding the TEK and VTN to the HP's SRK. The TEK is stored in the TVEM, which is a protected partition on the HP, thus it can be unencrypted only by the TVEM using the HP's TPM. The TEK will be protected in the TVEM and exposed on the HP only when requested by the TVEM for necessary operations.

The TEK is effectively "dual rooted" in both the host platform SRK and VTN root. This means that the TVEM cannot be cloned by copying its contents to another machine because the TEK is locked by the host's TPM. The TF maintains the VTN key and TEK root certificates and can revoke the VTN key or TEK at any time effectively removing privileges from TVEMs on rogue hosts.

TVEM migration is achieved by performing a TEK key migration from the current host platform to the new target platform. The TVEM migration is not direct to the target platform; it must go through the TF and verify that the target environment has the same level or greater trustworthiness than does the current host. Once the trustworthiness is determined, the TEK can be migrated to the new host by rewrapping the TEK with the new host's SRK. The TVEM and associated virtual machine can then be migrated to the new host without losing any information sealed by the TVEM.

6.3 Key Hierarchy and Management

The highest level key in a VTN is the master VTN key. All TEKs in the VTN are rooted and secured with the master VTN key. A master VTN key and certificate is generated for each VTN that the factory is responsible for managing. The VTN key is protected and stored with the TF platform's physical TPM SRK. The VTN root certificate along with the host platform SRK are used to generate all TEKs in the VTN.

The TF becomes the root authority for all VTNs under it auspices. TVEMs can be verified by checking their TEK certificates with the TF VTN authority. Since the TF is the root authority, it must maintain a key list of valid and revoked keys for each VTN. Once the VTN is deactivated, the VTN key is destroyed and all keys for that VTN must be revoked. A record of the revoked TEK should be kept to ensure that it will never be used again.

7 TVEM Design

The TVEM design is a set of functions grouped into five categories: legacy TPM functions, TVEM specific functions, storage functions, virtual environment interface, and user interface. Fig. 3 shows a block diagram of the high-level TVEM functions.

7.1 Legacy TPM Functions

The TVEM implements the following TPM functions per the TPM specification [8]: PCRs (as shadow registers), AIK, SRK, public key engine, secure hash, monotonic counter, and RNG.

The PCRs from the TPM are shadowed so that the virtual environment has the ability to read the configuration of the host platform. The virtual environment cannot modify the PCRs because it does not own the TPM. This is an important distinction for cloud computing environments. PCRs are written and modified by the hypervisor and host OS when virtual environments are launched. The configuration of the virtual environment is maintained separately in the *Virtual Environment Configuration Registers (VECRs).*

The RNG is an important construct for the virtual environment. Since virtual environments have limited ability to generate entropy, an external source for entropy is required. However, the TVEM itself is a virtual environment; therefore, it must also use an external source for entropy. The RNG for the TVEM needs to use the RNG on the host platform TPM to obtain the entropy required to generate encryption keys and nonces. The TPM's hardware based RNG generates sufficient entropy which the TVEM can use for the virtual environment. The latest Linux kernels (2.6.29 and above) support the TPM RNG, which can interface to a TVEM RNG in a VM.

The TPM hash function is implemented in the TVEM as a software module. Since the SHA-1 hash functions will be phased out in 2010 [18], another hash algorithm such as SHA-256 can be easily substituted.

The *Attestation Identity Key (AIK)* and proof is used to provide evidence that an entity is a valid TPM. The AIK is generated in conjunction with a trusted third party privacy authority in a manner that verifying the AIK established that the TPM is valid without revealing which specific TPM is validated. The AIK process can easily be converted to provide proof of a valid TVEM by simply adding the VTN factory's root certificate to the privacy authority's list.

There are also several TPM functions which may not be required at all. These include physical presence, physical maintenance commands (*e.g.*, field upgrade and set redirection) and other functions that are not needed for a software implementation.

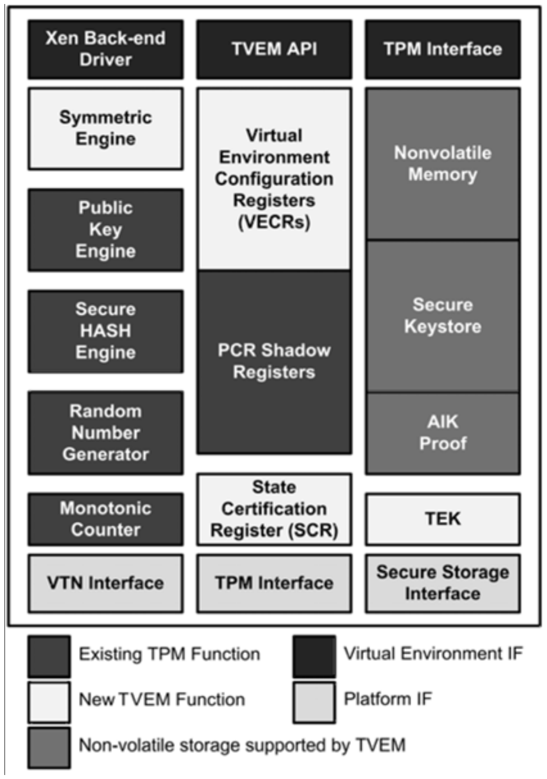

Fig. 3. The TVEM functional block diagram shows legacy TPM functions supported, the new TVEM functions, non-volatile storage, and host and platform interfaces

7.2 New TVEM Functions

Several new TVEM functions have been created to enhance the capability of the TVEM for virtual environments. These new functions include VECRs, the TEK, the *State Certification Register (SCR)*, and a symmetric encryption engine.

VECRs are equivalent to PCRs, and store configuration information of the virtual environment. There are 28 256 bit VECRs to support SHA-1; however, the VECRs can be configured up to 512 bits to support SHA-256. The VECRs are used for the virtual environment exactly as the PCR's for the physical platform. When a virtual environment is configured, the VECRs store configuration information about the virtual environment. The PCRs from the TPM are used; however, they are shadowed and only used for the purpose of determining the configuration of the host platform. The two sets of registers provide the ability to obtain configuration information about the platform and maintain a fine-grain detail about the configuration of the virtual environment. This enhanced view of both environments gives the virtual environment owner the ability to understand the security posture of the cloud.

The TEK is the endorsement key for the TVEM. The TEK functions exactly as the EK on a TPM, providing the master key for all TVEM functions and rooting all other keys. The TEK's dual trust root is essential to establishing trust in a virtual environment on a machine that is not owned by the information owner. Since the key is rooted both in the VTN and host platform, the key can be revoked by a change in configuration on either side thus invalidating the trust in the virtual environment.

A new register called the state certification register is used to verify that the state of the TVEM has not been modified. This feature will be described in a future paper that provides details on state medication and rollback prevention.

Another new feature of the TVEM compared to the VTPM is the addition of a symmetric data encryption engine. Cost is not a limiting factor and symmetric engines are very efficient in software; therefore, we can add an encryption engine and offload encryption tasks for small virtual environments. Providing an encryption engine allows smaller, hardened environments instead of bloated OSes. Additionally, export controls are not a major concern with a TVEM. If a TVEM is to be exported outside of its originating country, the encryption engine can be easily removed or swapped with an engine without export controls. Finally, the encryption engine can provide enhanced security by ensuring that correct and verified implementations are used.

7.3 Non-volatile Storage

To support operation of the TVEM across multiple sessions and migration, the information placed in the non-volatile storage of the TVEM must be persistent. To make the non-volatile memory persistent, the contents of the memory are backed up to the TF where they are stored until the TF is terminated. Each TVEM's non-volatile memory image must be maintained until the TVEM is terminated.

To maintain the image of each TVEM's non-volatile memory, every time the non-volatile memory in the TVEM is updated the TF updates its backup image. The update is done by sending a message with the contents to the TF over the host platform secure storage interface. When the TF receives the message, it updates its backup image for the specified TVM instance accordingly.

An additional benefit of keeping the backup image of the non-volatile memory is the ability to verify the local TVEM image. The TVEM may send the full contents of the TF at any point to request a verification of the content. The TF can compare the sent image with its backup image and verify that both are identical. If the content of the TVEM's memory were tampered, the comparison would be different.

7.4 Virtual Environment Interfaces

One of the biggest differences between a VTPM and a TVEM is the virtual environment interfaces. TVEM virtual environment interfaces on the VM side of the TVEM are designed to accommodate the many types of virtual environments that may need TVEM services. Not all environments will have a full TSS or cryptographic API; therefore, an API on the TVEM provides the cryptographic services for the virtual environment.

The TVEM has three unique interfaces to the virtual environment as shown in Fig. 3: a Xen back-end driver, a TVEM API, and a standard TPM interface for compatibility with TPM 1.2. Each virtual environment that requires a TVEM service has the option to choose the interface it will use. A virtual environment may use multiple interfaces if desired. For example, the OS may use the Xen driver in the kernel and an application may use the TVEM API.

7.5 TPM Interface

The TPM interface will work with any program written to support a TPM 1.2. This interface provides backward compatibility for the TVEM where applications are expecting a TPM or VTPM. The TVEM can appear to be a valid TPM and operate as a TPM replacement.

Note that when accessing a TVEM as a TPM, the VECRs are accessed instead of the PCRs. The VECRs represent the state of the virtual environment, which is the context that the guest is operating.

TVEM API. The TVEM API provides applications a direct interface to the TVEM through a set of function calls. This interface allows OSes and applications without a full TSS or cryptographic library the ability to use the TVEM easily.

The API provides access to extended TVEM functions including the symmetric encryption engine and PCR shadow registers. The interface can also be extended to connect with additional cryptographic hardware such as smartcards and biometrics as well.

Xen Backend Driver. The Xen back-end driver will interface directly to a Xen kernel front-end driver [19]. This capability enables any virtual environment running on a Xen hypervisor the ability to interface to TVEM with the simple addition of the front-end driver to the virtual environment. The Xen interface is an extension of the API and provides a seamless interface for the virtual environment.

7.6 Host Platform Interfaces

The host platform interfaces are to the host platform side of the TVEM. The TVEM interfaces to the host platform differently than a VTPM. The TVEM uses the TPM and host platform as a service to provide functions required for secure operation. The TVEM uses the host interfaces for host TPM services, communicating with the VTN, and for storing non-volatile information.

Host TPM Interface. The TVEM communicates with the host TPM via the TVEM manager. The TVEM manager is responsible for arbitrating access to the TPM. All requests that require host TPM access use this interface including the reading of the TPM's PCRs, storing and retrieving keys to the TPM, and accessing the RNG.

VTN Interface. The VTN interface is used to manage keys and communicate with the TF over the encrypted VTN.

Secure Storage Interface. The secure storage interface is a VTN interface used by the TVEM to store non-volatile memory securely. The secure storage interface uses the VTN to send all non-volatile writes to the TF for backup storage. On provisioning of a TVEM, the secure storage interface is used to populate the non-volatile storage areas on the TVEM from the backup image. The secure storage interface is also used to verify the contents of the non-volatile memory with the backup on the TF.

8 Discussion

The TVEM provides many advantages over a VTPM in a cloud computing environment. The management of TVEM from the TVEM factory provides the ability to control and monitor TVEMs in a VTN and provides enhanced situational awareness to the information owner. The TVEM also provides system designers and information owners support for everything from simple single purpose applications to full OSes. The virtual environment specific functions enable ease of use, and the modular design enables flexibility for deployment. TVEMs provide strong cryptographic support for securing a virtual environment on a cloud host platform. The unique dual rooted key structure provides flexibility to maintain trust in the virtual environment and allows information owners to control the confidentiality of their data on the host platforms.

TVEM configurability is another advantage over VTPMs. By allowing information owners to customize their protection requirements, they have flexibility to use cloud computing services that were previously unavailable.

TVEM improves security in our example web server application by ensuring that the environment is executing on a trustworthy platform and is correctly configured. As the RTS, the TVEM protects the server SSL and EV certificates by encrypting the keys with a unique SRK and storing them in persistent non-volatile memory. For stronger protection, the TVEM can bind the keys to the configuration of the host platform and/or virtual environment. TVEM also provides a high entropy source for random number generation for SSL sessions.

It is important to remember that TVEMs are not designed to defend against hardware based attack. TVEMs are software devices and any attacker with access to certain ports (*e.g.*, PCI, IEEE 1394 FireWire), hardware monitoring devices, emulation and debug equipment, or memory inspection equipment can circumvent the TVEM's security. Since hardware attacks cannot be detected or defended against, physical security of cloud datacenters is of utmost importance.

Another type of attack that TVEM cannot defend against is a dishonest host or service provider. The information owner is at the mercy of the service provider to provide the services agreed upon in a service agreement. If the host platform lies and falsely reports its attestation values to the TF, the TF has no basis for challenging the integrity of the platform. To prevent the dishonest host, social trust must be used as it is likely that once it is detected that the host is falsely reporting, word of the dishonesty will be spread through the community and the service provider's reputation will diminish.

9 Conclusion

TVEM is a new and unique concept for rooting trust in the cloud. The TVEM solves the problem of rooting trust in IaaS cloud computing where a service provider owns the platform on which an information owner's virtual environment is operating. TVEM enhances security by allowing for trust in the virtual environment that is distinct and separate from the hosting platform. The TVEM protects information and conveys ownership in the cloud through the TEK generation process, which creates a dual rooted trust for the virtual environment. This dual rooted trust is necessary to accommodate the unique relationships that occur in cloud computing.

The TVEM gives information owners control of their sensitive and private data in the cloud by providing assurance that their environments are correctly configured and data are kept confidential. The TVEM provides management control of trust through the centralized TVEM factory control facility, key hierarchy, and modular configurable architecture.

This paper introduces the high level system architecture and design concepts of a necessarily somewhat complex TVEM system. The definitions of the TVEM, TVN, and TEK provided here are strong building blocks to continue developing the details of the sub-modules and components. To ensure the TVEM meets the needs of the cloud computing users, the TVEM system should go through a formal specification development cycle with representatives from many stakeholders, including providers, customers, trusted computing experts, and cloud computing researchers. With proper vetting and industry support, the TVEM can be a valuable security component for IaaS cloud computing, enabling a higher adoption rate and a more secure cloud.

The TVEM provides information owners the ability to control their information in the cloud and to realize the savings and benefits that come from the economies of scale that the cloud provides. When combined with other cloud computing security technologies such as Private Virtual Infrastructure and Locator Bots, TVEMs enable a powerful solution to protecting information in cloud computing.

References

1. Carr, N.G.: The Big Switch: Rewiring the World, from Edison to Google. W.W. Norton & Company, New York (2008)
2. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: Above the Clouds: A Berkeley View of Cloud Computing. University of California, Berkeley (2009), `http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf`
3. Krautheim, F.J., Phatak, D.S., Sherman, A.T.: Private Virtual Infrastructure: A Model for Trustworthy Utility Cloud Computing. TR-CS-10-04. University of Maryland Baltimore County, Baltimore, MD (2010), `http://www.cisa.umbc.edu/papers/krautheim_tr-cs-10-04.pdf`
4. Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M.: A Break in the Clouds: Towards a Cloud Definition. ACM SIGCOMM Computer Communication Review 39, 50–55 (2009)
5. Grawrock, D.: The Intel Safer Computing Initiative. Intel Press, Hillsboro (2006)
6. Scarlata, V., Rozas, C., Wiseman, M., Grawrock, D., Vishik, C.: TPM Virtualization: Building a General Framework. In: Pohlmann, N., Reimer, H. (eds.) Trusted Computing, pp. 43–56. Vieweg+Teubner, Wiesbaden (2008)
7. Krautheim, F.J.: Private Virtual Infrastructure for Cloud Computing. In: Workshop on Hot Topics in Cloud Computing, San Diego, CA (2009)
8. TPM Specification Version 1.2 Revision 103. Trusted Computing Group (2007), `http://www.trustedcomputinggroup.org/resources/tpm_main_specification`

9. Abramson, D., Jackson, J., Muthrasanallur, S., Neiger, G., Regnier, G., Sankaran, R., Schoinas, I., Uhlig, R., Vembu, B., Wiegert, J.: Intel Virtualization Technology for Directed I/O. Intel Technology Journal 10, 179–192 (2006)
10. Intel Trusted Execution Technology, `http://www.intel.com/technology/security/`
11. Berger, S., Cáceres, R., Goldman, K.A., Perez, R., Sailer, R., van Doorn, L.: vTPM: Virtualizing the Trusted Platform Module. In: Proceedings of the 15th USENIX Security Symposium, Vancouver, BC (2006)
12. England, P., Loeser, J.: Para-Virtualized TPM Sharing. In: Lipp, P., Sadeghi, A.-R., Koch, K.-M. (eds.) Trust 2008. LNCS, vol. 4968, pp. 119–132. Springer, Heidelberg (2008)
13. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the Art of Virtualization. ACM SIGOPS Operating Systems Review 37, 164–177 (2003)
14. Sadeghi, A.-R., Stüble, C., Winandy, M.: Property-Based TPM Virtualization. In: Wu, T.-C., Lei, C.-L., Rijmen, V., Lee, D.-T. (eds.) ISC 2008. LNCS, vol. 5222, pp. 1–16. Springer, Heidelberg (2008)
15. Strasser, M.: A Software-based TPM Emulator for Linux. Department of Computer Science, Swiss Federal Institute of Technology, Zurich (2004)
16. Wang, X., Yin, Y.L., Yu, H.: Finding Collisions in the Full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
17. Sailer, R., Valdez, E., Jaeger, T., Perez, R., van Doorn, L., Griffin, J.L., Berger, S.: sHype: Secure Hypervisor Approach to Trusted Virtualized Systems. IBM, Yorktown Heights, NY (2005), `http://www.research.ibm.com/secure_systems_department/projects/hypervisor/`
18. Dang, Q.: Recommendation for Applications Using Approved Hash Algorithms. NIST Special Publication, vol. 800. NIST, Gaithersburg (2009)
19. Chisnall, D.: The Definitive Guide to the Xen Hypervisor. Prentice Hall, Upper Saddle River (2008)