Security for the cloud infrastructure: Trusted virtual data center implementation

S. Berger
R. Cáceres
K. Goldman
D. Pendarakis
R. Perez
J. R. Rao
E. Rom
R. Sailer
W. Schildhauer
D. Srinivasan
S. Tal
E. Valdez

The trusted virtual data center (TVDc) is a technology developed to address the need for strong isolation and integrity guarantees in virtualized environments. In this paper, we extend previous work on the TVDc by implementing controlled access to networked storage based on security labels and by implementing management prototypes that demonstrate the enforcement of isolation constraints and integrity checking. In addition, we extend the management paradigm for the TVDc with a hierarchical administration model based on trusted virtual domains and describe the challenges for future research.

Introduction

Virtualization technology is used increasingly in data centers, both for commodity and high-end servers. Among the primary drivers for this trend is the ability to aggregate multiple workloads to run on the same set of physical resources, thus resulting in increased server utilization and reduced space and power consumption.

Virtualization utilizes a software layer, called virtual machine monitor (VMM) or hypervisor, to create virtual machines (VMs) that run concurrently on the same physical system. The ease of creating, migrating, and deleting VMs is responsible for the new flexibility in server deployment and new workload mobility. The consolidation of computing resources is facilitating the emergence of the cloud-computing paradigm, in which information technology (IT) infrastructure, applications, and data are provided to users as services over a network, public or private. Computing clouds can achieve economies of scale, thus providing services that can be made available globally, at a very large scale and low cost.

However, placing different customers' workloads on the same physical machines may lead to security vulnerabilities, such as denial of service attacks, and possible loss of sensitive data. Furthermore, an IT infrastructure that consists of a large number of heterogeneous components involves complex configuration management tasks, which increases the likelihood for misconfiguration, an additional source of increased vulnerability.

The trusted virtual data center (TVDc) [1] is a technology developed to address the need for strong isolation and integrity guarantees in virtualized, cloud computing environments. VMs and associated resources are grouped into trusted virtual domains (TVDs) [2, 3]. A TVD is a security domain that uniformly enforces an isolation policy across its members. The policy specifies which VMs can access which resources and which VMs can communicate with each other. For integrity guarantees, the TVDc leverages a virtualized root of trust to determine the identity and integrity of the software loaded on VMs. The TVDc prototype in Reference [1] has demonstrated the feasibility of managing TVDs across servers, networks, and storage resources.

The goal of TVDc is to isolate customer workloads from each other. In particular, the TVDc aims to: 1) prevent data from leaking from one customer workload to another, even when a VM running the workloads malfunctions; 2) ensure that viruses and other malicious code cannot spread from one customer workload to another and that break-ins in one workload do not threaten the workloads active within the same physical resource; and 3) prevent or reduce the incidence of failed configuration management tasks (i.e., misconfiguration).

In this paper we extend the prototype described in Reference [1] by implementing controlled access to networked storage based on security labels and by implementing management prototypes that demonstrate the enforcement of isolation constraints and integrity

0018-8646/09/$5.00 © 2009 IBM

checking. In addition, we extend the management paradigm for TVDc with a hierarchical administration model based on TVDs and describe the challenges for future research.

This paper is organized as follows. The next section describes the TVDc isolation policy and TVDc management principles, which cover isolation management and integrity management. The section "TVDc implementation" describes the components for realizing TVD in data centers and the associated networks and networked storage. The section "TVDc management application prototypes" describes two prototypes that demonstrate how isolation and integrity constraints are enforced. Finally, the section "Conclusions and future work" summarizes our results and lists some of the challenges that remain in applying and extending the TVDc concept.

TVDc principles

TVDc provides isolation of both systems management and customer workloads by employing an isolation policy and different types of workload isolation mechanisms. The isolation policy abstracts the physical infrastructure and allows for automated policy-driven configuration management of data center resources, such as platform access and network connection. Below, we discuss the isolation policy and supported isolation on infrastructure components and the management of TVDc to support isolation and integrity.

TVDc builds on existing components to provide isolation. These components include role-based access control, hypervisor-based isolation, and protected communication channels such as virtual local area networks (VLANs, IEEE standard 802.1Q** [4]). Additionally, TVDc employs the Trusted Computing Group (TCG) load-time attestation mechanism to verify the software integrity of systems [5].

Isolation policy and enforcement

The TVDc isolation policy is coarse-grained because its unit is the TVD, the set of VMs and associated resources that serve a common purpose. The boundaries of a TVD are defined by labeling all VMs and associated resources within the TVD with a unique TVD identifier known as a *security label* (also referred to as a *color*). For example, a TVD label can denote a customer (e.g., IBM) or a customer workload (e.g., IBM accounting).

The TVDc isolation policy has two parts: (1) the label definitions, which define security contexts that can be assigned to VMs and resources (i.e., TVD); and (2) the anti-collocation definitions, which enforce restrictions on which VMs can run on the same system at the same time based on the TVD to which they are assigned. The access control management defines system administration roles

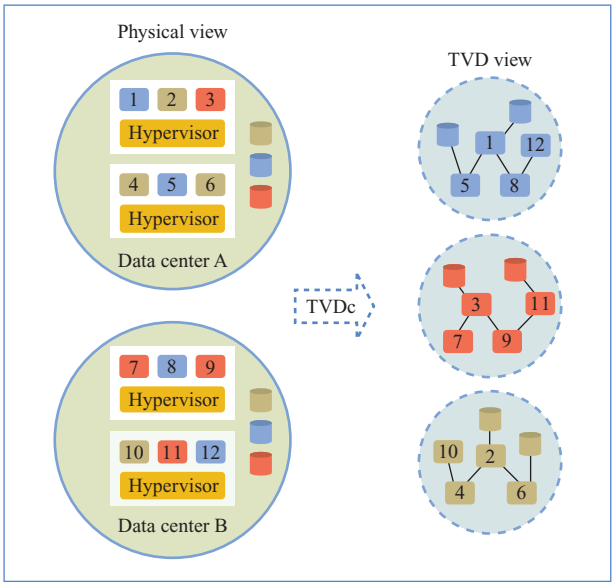

Figure 1

Trusted virtual domain (TVD) view of physical data centers and their resources.

and assigns permissions that are based on the security labels. Thus, the integration of TVDc isolation policies and role-based access control management policies is implemented by connecting the security labels of the TVDc policy with the permissions assigned to roles in the management.

Security labels can be thought of as entitling memberships to TVDs, with each TVD being identified by its security label, as shown in **Figure 1**. The labeling of VMs, resources, and roles can be viewed as coloring (we use the terms *color* and *security label* interchangeably). Figure 1 shows two physical data centers and their resources, such as servers, storage, and network components. The TVD view on the right captures the association of physical resources with security labels (colors). For example, we see in the figure the blue domain, which includes VMs 1, 5, 8, and 12, and associated storage.

The following describes the different kinds of isolation supported on the workload and administration planes:

Data sharing—In this color model, VMs can share data through network, storage, or direct sharing (e.g., shared memory) if they share a common color. The color model is suitable for interaction that is intrinsically bidirectional. It does not distinguish between *read* and *write* operations between subjects and objects; in our case, a subject is a VM and an object can be a VM or a resource. This sharing model is similar to the nonhierarchical-type enforcement security model [6].

VMM system authorization—The colors are also used to restrict which workloads (TVD identifier) a VMM system can run. A VMM can start a VM only if the colors contained in the VMM system label are a superset of the colors assigned to a VM. This mechanism offers a simple means to partition the physical infrastructure for stronger isolation of certain workloads.

Collocation constraints—Anti-collocation rules of the policy restrict which VMs can run simultaneously on the same VMM system. Each rule describes a set of colors that conflict. For example, if the set of colors in the rule includes red and blue, a red VM and a blue VM may not run at the same time on the same system.

Management constraints—To integrate TVDc VM isolation and management isolation, we assign the TVD colors to roles (as permissions) and roles to administrators. This confines administrators to managing resources and VMs that are labeled with a subset of the colors that are assigned to their role.

In our TVDc implementation, sharing, system authorization, and collocation constraints are enforced by the VMM, storage, and network infrastructure. Management constraints are enforced by the overarching virtualization management solution.

TVDc management

TVDc shifts the focus of security from independent systems and tools to collections of complete systems and hardware resources working together to provide services to workloads within a data center using virtualization. Essential to this computing environment is a central point of administration to define the TVDs, to assign labels to physical and virtual resources, to deploy security policies, and to consolidate the evaluation of measurement data from hypervisor and guest VMs for integrity analysis.

A data center administrator may create virtual environments across the physical IT resources that not only perform intended services, but can also be viewed logically as data centers in their own right. High-level security and operation policies can be mapped to the components of the virtual data centers to provide strong isolation (i.e., containment) and trust establishment (i.e., integrity) for software workloads consisting of one or more VMs and their respective resources.

Isolation management

Isolation management allows systems administrators to define and administer security policies for containment and access control both within and across virtual domains. Once the security labels are created and assigned, policies enabled on the management VMs allow VMs that have the same label to form coalitions of cooperating resources (for each hypervisor, a VM—known as a *management VM*—is usually created to help configure and allocate resources to other VMs). However, policies can also establish mutual exclusion, such that VMs of different colors cannot be run simultaneously on the same system.

A systems management application implementing isolation management must not only provide a centralized location to define and distribute security policies and assign resources, but must also furnish multiple, hierarchical roles of administration and different user interface views for each role. These roles include the IT data center administrator, the TVDc administrator, and, optionally, the tenant administrator, as shown in **Figure 2**.

At the top level of the hierarchy is the *IT data center administrator*, who manages all the resources in the data center using the systems management application. The IT data center administrator: 1) discovers the physical resources in the data center; 2) discovers the virtual resources, including management VMs; and 3) groups the discovered resources into TVDcs.

In the example shown in Figure 2, the IT data center administrator has partitioned the physical resources into TVDc A and TVDc B. Each TVDc has an administrator who is responsible for it. Figure 2 also shows that a TVDc can consist of multiple TVDs, and each TVD has its own administrator, known as the *tenant administrator*. Once the physical resources are assigned to the TVDcs, the IT data center administrator creates and deploys isolation policies to the management VMs, creates security labels, and delegates the isolation management of resources and VMs within the TVDc to *TVDc administrators*. The use case steps are to:

- Define security labels to be used by the workloads within a TVDc.
- Define the collocation constraints; e.g., which workloads should never be collocated on the same hardware (physical constraint) or run at the same time on the same system (run-time constraint).
- Use the definitions from steps 1 and 2 to create a policy for the targeted hypervisor and to deploy it to the management VMs.

Note that a single isolation policy is deployed within a particular TVDc. This central policy is independent of the physical platform or the hypervisor and ensures consistent isolation across heterogeneous platforms. The enforcement of the policy on the individual systems depends on the hypervisor and the platform.

After policy deployment to the TVDc, the IT data center administrator authorizes the next administrator in the hierarchy, the TVDc administrator, to 1) access the

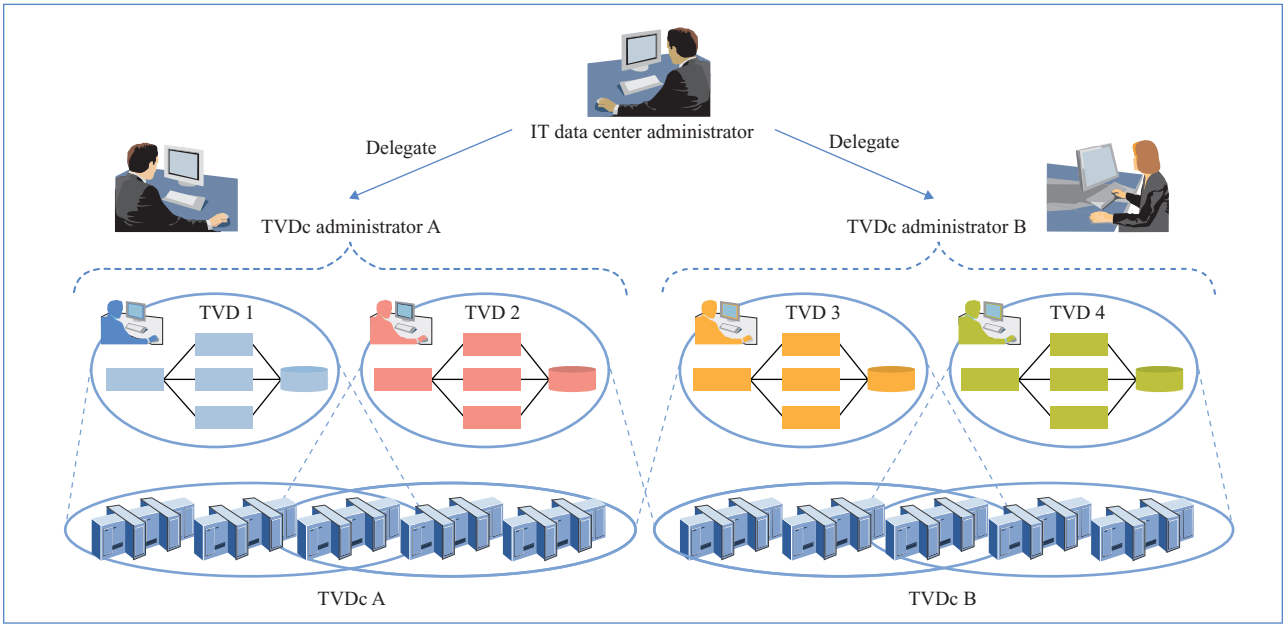

Figure 2

Administration domains in the data center.

created TVDc and 2) assign security labels to resources in the TVDc. The TVDc administrator assigns the security labels to the hypervisor and the management VMs, which enables them to enforce the sharing or exclusion of resources as required.

Whereas the IT data center administrator and TVDc administrator have access to the full set of resources assigned to a TVDc, it may be beneficial to create an additional administrative role, known as a *tenant administrator*, whose job is limited to managing the resources in a particular TVD. This grants the tenant administrator the privileges needed to control the assigned workload, but not the ability to compromise isolation by tampering with the isolation policy or labels. As shown in Figure 2, the blue TVD is managed by the blue tenant administrator, and so on.

Whereas the management roles are, by default, strictly separated from each other, roles can be combined (collapsed). For example, the TVDc administrator may enable a tenant administrator to manage more than one TVD, as in multiple, isolated workloads running on behalf of the same customer.

Integrity management

The TVDc can be used to establish trust in a remote computer by verifying the integrity of the software loaded on that computer, whether it is a physical or virtual system.

In addition to measurements of software components taken during the boot process [7], as specified by the TCG, the Integrity Measurement Architecture (IMA) [8] extends the list of measured components to files loaded into memory for execution. Systems management applications can provide integrity attestation by comparing the list of measurements in the management VM or guest VM with known values to ensure its proper operation and conformance to local update policies.

All measurements of the management VM or guest VM are taken and provided using integrity infrastructure components described in the section "Virtualized TPM"; these components include the TPM as specified by the TCG and the virtualized TPM.

The systems management application is a challenger party in the remote attestation of management VMs, or guest VMs, or the VMM through the management VM. Such attestations can be invoked on demand by the IT data center administrator or the TVDc administrator, scheduled periodically, or monitored in real time. In addition, these administrators may specify remediation actions to be taken when events indicating out-of-policy measurements are received.

Integrity attestation requires a database of reference measurements that can be compared with run-time measurements from VMs. The IT data center administrator assembles these measurements for the supported operating systems, their updates, packages that

may not be part of the core distributions, and files that are not executable, such as configuration files or files containing the isolation policies installed in the VMM and the management VMs. As with isolation management, the updating of the integrity management database, which is performed by the IT data center administrator, is a privileged operation and must not be accessible to a TVDc administrator. However, TVDc administrators may read or subscribe to integrity measurement results for verification and remediation.

In addition to defining and managing configurations related to integrity management, other administrative information may be accessible to both the IT data center administrator and the TVDc administrator, though the latter may view only those resources within the TVDc. This information may include the following:

- Overall compliance of the data center, to determine trends and problem areas.
- Summary of VMs that have unknown or out-of-policy measurements.
- Lists of measured files on a particular VM.
- Events indicating unknown or out-of-policy measurements.

TVDc implementation

We implemented a TVDc prototype in a data center populated with IBM BladeCenter* [9] units mounted on racks. Each rack holds a fixed number of server modules, called *blades*. Each blade is a standalone computing system with processors, memory, and network and storage adapters. The blades are interconnected using rack-mounted Ethernet switches and have access to network storage, such as Fibre Channel-based storage area networks (SANs). The rack also contains systems management tools and provides power and cooling for all the rack modules.

Figure 3 shows a simplified view of a TVDc implementation consisting of two TVDs, labeled blue and red, with separate logical (virtual) networks and storage devices. The set of physical resources include racks labeled Rack 1 and Rack 2, each of which contains a number of blades. The blades in the foreground are labeled Blade 1 and Blade 2. The dashed arrows from the data center administrator icon to blades, physical switches, and SAN volume control (SVC) nodes represent paths used by the management application to configure these components.

The administrator configures the Xen** systems running on Blades 1 and 2 using Xen API, the application programming interface (API). Configuring of the depicted implementation includes labeling (coloring) resources, such as VMs and VLANs, either blue or red. The figure also shows a special VM in each Xen system labeled Domain 0. The Domain 0 VM provides TPM functionality by way of a virtualized TPM (vTPM) process that runs a dedicated vTPM instance for each of the remaining VMs running on the Xen system. Domain 0 also supports a software bridge for each TVD, shown in the figure by entities labeled Red bridge and Blue bridge. In each Xen system, all VMs of the same color are connected to the appropriate bridge. That is, all blue VMs are connected to the blue bridge and all red VMs are connected to the red bridge. The bridge thus enables VMs of the same color to communicate.

The solid red and blue lines in Figure 3 from the Xen systems to the external switches represent VLANs (configured on the switch ports) that connect the blades to the external network. The solid colored lines between the switches represent connections between the racks that support network traffic flows (blue and red) between the racks. This allows, for example, blue VM 1 on blade 1 to communicate with blue VM 4 on blade 2, located on a different rack. Finally, the figure shows SVC with colored storage volumes where access to storage volumes is permitted only for the traffic of the same color (that is, red VMs can only access red storage volumes).

Our TVDc implementation employs mandatory access control in the VMM infrastructure, using the Xen open-source hypervisor [10] extended with the sHype security architecture [11]. For network isolation, we configure internal rack Ethernet switches to support the deployed isolation policy. For network storage, we extend a capability-based access control mechanism to permit access based on TVDc security labels. Additionally, we provide TCG TPM functionality to VMs running on those virtualized systems.

Hypervisor

The TVDc prototype employs the sHype hypervisor security architecture to isolate between different sets of VMs on a system. sHype depends on the core hypervisor (Xen) isolation property to ensure isolation of VMs and resources on a virtualized system. In brief, sHype supervises the sharing of resources among VMs as well as inter-VM communication according to the deployed isolation policy on the system. sHype implements mandatory access control as a reference monitor [12] inside the hypervisor. See Reference [11] and Reference [13] for additional details.

Network isolation

The basic idea behind implementing TVDc network isolation is to associate with VLANs the same kind of label that we associate with VMs. We restrict VMs to VLANs with matching security labels. This way, VMs

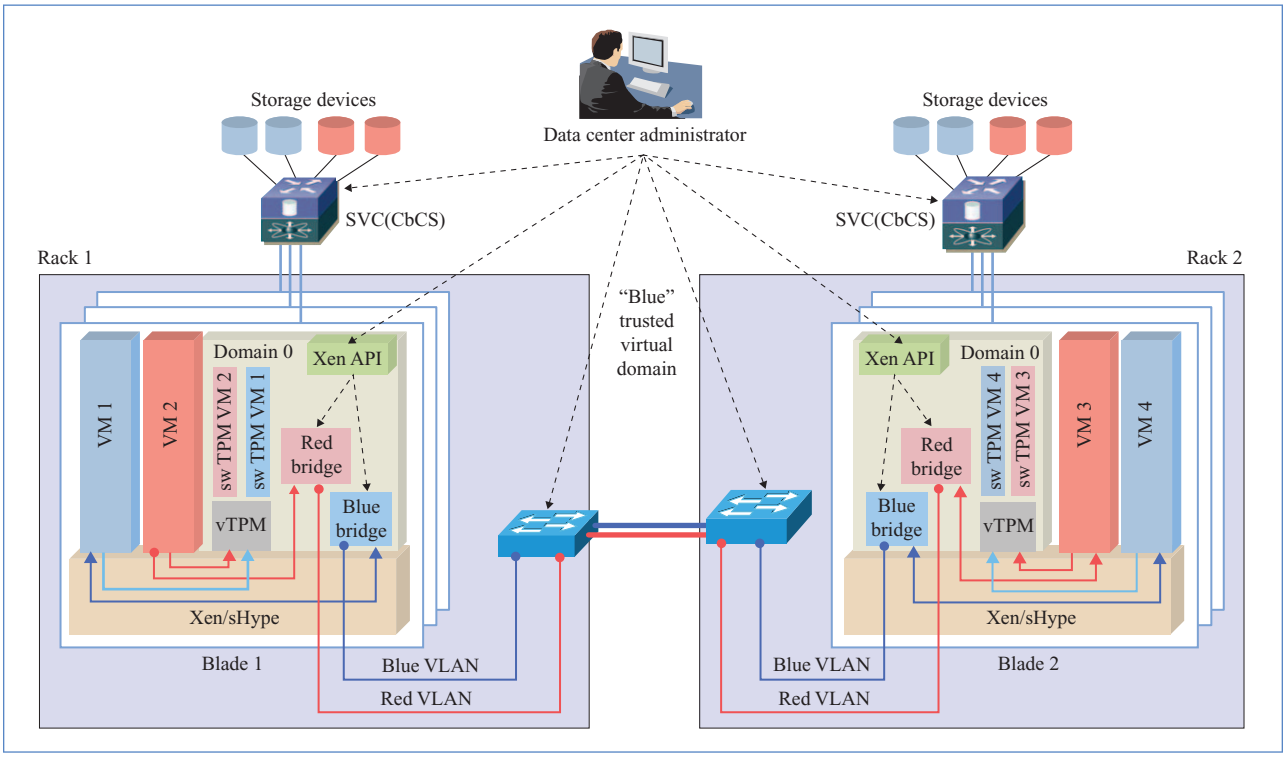

Figure 3

TVDc implementation.

with the same security label can communicate through the VLAN but cannot communicate with any other VM.

To enable network isolation, we extend our TVDc facility to control VM access to LANs using VLANs. VLANs emulate separate physical LANs on a single physical LAN by prepending to each Ethernet frame the appropriate VLAN ID and maintaining strict separation of frames with different IDs.

In our implementation, the TVDc network isolation component configures VLANs internally within virtualized systems and externally on rack Ethernet switches that connect virtualized systems (blades) to the network. Thus, VMs can communicate only with other VMs that have the same security label on the same local system or on remote systems.

On Xen systems, Domain 0 owns the physical network device and controls network access to the guest VMs on the system. The TVDc network isolation component, which configures the appropriate VLANs within Domain 0, creates a software-emulated Ethernet bridge for each VLAN that a Xen system is authorized to access. When a guest VM is created, it is given access to the bridges for the VLANs that it is authorized to access. This allows two guest VMs that have the same security label on the same Xen system to connect to the same software bridge and thus communicate, whereas guest VMs that do not have the same security label cannot communicate even if they are on the same physical system. The TVDc network isolation component also configures rack Ethernet switches to connect each Xen system only to those VLANs the system is authorized to access following the least-privilege security principle.

Networked storage isolation

To support storage isolation based on security labels, we integrated into the TVDc prototype the Capability-based Command Security (CbCS) mechanism [14]. CbCS is a capability-based extension of the SCSI (Small Computer System Interface) protocol for access control to networked storage devices. The protocol requires that storage I/O commands initiated by any client, such as a guest VM, provide a cryptographically hardened credential. The credential is obtained from a security policy manager and contains a capability that encodes the rights the client has to access the storage volume (e.g., read and write or read-only access rights).

The storage security manager cryptographically hardens the credential with a message authentication code

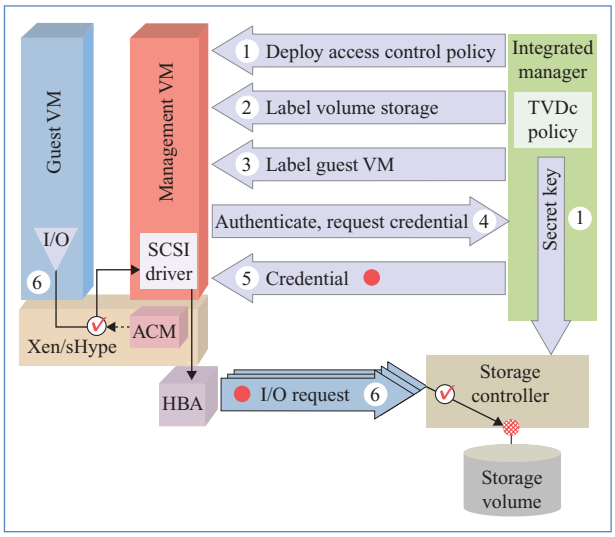

Figure 4

TVDc and CbCS integration.

(MAC) using a symmetric key shared with the storage device so that credentials cannot be forged or modified. The storage device validates the authenticity of the credential and grants or denies access on the basis of encoded capability. Our TVDc implementation has an additional check to credential generation, namely a client (i.e., a guest VM) and the storage volume must have the same security label in order for the client to obtain a credential to access the storage volume.

Figure 4 illustrates the integration of the CbCS mechanism into the TVDc prototype. The integrated TVDc management application in the figure combines both TVDc and storage security management features. In the first step, the management application deploys the access control policy to the management VM (Domain 0 on Xen systems) and sets the symmetric secret key to the storage controller. In the second step, the storage volume resource is labeled. In the third step, a guest VM is created and assigned a security label and a storage private key that is later used for authentication to the management application. This key, as well as VM configuration information, is kept by the management VM. In the fourth step, the management VM authenticates the VM to the management application and asks for the credential for the guest VM to access the storage volume. On the management application, if the VM and the storage volume have matching labels, a credential is returned; otherwise, the management application fails the request. Finally, when the guest VM sends an I/O command, it is validated both by the hypervisor checking for matching security labels and by the storage system checking the validity of the credential before permitting the command to be executed according to the specified capability.

In our implementation, the TVDc management application, the storage systems, and the management VM are part of the trusted computing base. The TVDc management application is trusted to safely store long-lived keys and compute access controls correctly according to the isolation policy. Additionally, guest VMs trust Domain 0 not to leak their private keys and credentials. The storage system denies access to storage if the CbCS client does not follow the CbCS protocol.

To mitigate security threats, we employ the principle of least privilege with respect to allocating access to storage. When a guest VM is migrated out of a system, the management VM discards the authentication information and any credentials associated with that guest VM. Therefore, access to a host system is limited to the guest VMs currently running on it.

Virtualized TPM

The virtualized TPM (vTPM) [15], a software emulation of the TPM specification [16] from TCG, provides TPM functionality to guest VMs running on virtualized hardware platforms. The vTPM can run in a management VM, within the hypervisor itself, or in secure hardware such as the IBM 4764 PCI-X Cryptographic Coprocessor [17]. As with any virtualized hardware, the virtualization is transparent to the guest VM, and applications using the vTPM run unmodified. Each guest accesses a private, unique, and complete vTPM instance.

The software emulation of the TPM adds several extensions for virtualization management, which a guest VM would not normally use. The extensions fall into several categories:

- Creating and deleting virtual TPM instances.
- Provisioning (e.g., enabling and activating the TPM, creating the TPM endorsement key, adding early boot measurements).
- Securely migrating the TPM instance when a guest is being migrated to another physical platform.

When provisioning a vTPM for integrity quoting, a signing key certificate must be created. Several approaches are discussed in Reference [15], including:

- Using the hardware platform TPM as a local certificate authority for the vTPM endorsement key (EK).
- Using the hardware platform TPM as a local certificate authority for the vTPM quote signing key.
- Using a local data center certificate authority to directly certify the vTPM EK.

- Hosting the vTPM in a secure coprocessor such as the IBM 4764, and using the coprocessor keys and certificate to certify the vTPM EK.

vTPM integration with virtualization management

To make TPM functionality available to VMs, we integrated the vTPM life-cycle management into the system management software. Users of a virtualized system have the flexibility to configure selected VMs with TPM support. We enabled the addition of TPM functionality to a VM similar to that of other devices, such as a network adapter or a hard disk partition, by letting the users specify these devices in the VM configuration. However, unlike advanced hot-plugging operations that may be available for network or disk adapters, the vTPM may not be removed or attached while the VM is running. We disallow these operations because a measurement program running inside an OS relies on the constant availability of a TPM so that an attestation can return proper results at any given time.

Furthermore, the binding of a VM to its vTPM must be permanent because a TPM contains persistent data that may hold secrets such as private keys, which must be protected from access by other parties. The association of a vTPM instance with its VM is stored in a table that holds VM-to-vTPM instance number mappings. The VM itself has no control over which vTPM instance its TPM requests are routed to; the management VM determines the association.

To validate our design, we integrated the vTPM into the Xen hypervisor environment and the low-level management software (*xend*) that runs on Domain 0. We extended the management application programming interface (API) offered by *xend* through remote procedure calls based on XML-RPC (Extensible Markup Language Remote Procedure Call) with functions that enable the life-cycle management of the private TPM of a VM. A simple operation such as attaching a TPM to a VM provides the basic functionality necessary to make a TPM available to an OS inside a VM. More advanced operations, such as the migration of the vTPM instance alongside its VM, are triggered by VM management software.

TVDc management application prototypes

Managing TVDc-supported data centers is a topic of ongoing research. We describe experimental systems management prototypes we developed that implement isolation constraints and integrity checks.

Authorization labels and security labels

In support of TVDc deployment, we have implemented TVDc-top, a simple Python [18] application that enables the visualization of the security features we added to the Xen hypervisor. **Figure 5** shows a screenshot of TVDc-top display involving six VMs deployed on two blades, cso183 and cso195. Policy identified as *sHype-policy* is deployed on both blades and each VM is labeled either Blue or Red. The authorization (system) label of both blades has been set to SystemManagement; this allows red and blue VMs to run on the blades. The figure also shows that VMs of the same color are connected to the same VLAN; blue VMs are connected to VLAN 100, and red VMs are connected to VLAN 200.

TVDc-top also supports basic VM life-cycle operations, such as start, stop, suspend, and resume, as well as operations to migrate VMs from one physical machine to another. It also implements security-related functionality, such as deploying an sHype security policy to or from a blade. TVDc-top has served as a useful tool for testing sHype security extensions and for demonstrating the functions implemented in the prototype.

Integrity management

We have implemented integrity management in a prototype based on IBM Systems Director 5.20.1 [19] as the management server. The VM environment is based on Xen, with vTPM running in Domain 0. When a guest VM is started, it gets access to its own vTPM instance. The Director agent running in each VM has been enhanced to retrieve integrity measurements from the vTPM instance. These integrity measurements are exposed by the vTPM to the guest VM through the sysfs file system. To verify the integrity of a guest VM, the Director management server retrieves (from the Director agent) integrity measurement data collected since the guest VM boot time and compares such data to the reference data in its master database of valid measurements. If the comparison shows a discrepancy, the Director server shuts down the guest VM immediately and notifies the administrator; otherwise, it simply adds an entry to its event log. This prototype was successfully demonstrated at the Intel Developer Forum 2007 in San Francisco [20].

Related work

To provide trust and containment guarantees to the data center environment, the TVDc concept integrates virtualization-based security and systems management. TVDc is an implementation of the TVD abstraction [2, 3, 21, 22], which aims to simplify user and administration interactions on large-scale systems by offloading the security enforcement onto the infrastructure. TVDc builds on previous work on trusted computing [5], vTPM [15], trust measurement architecture [8, 23], and sHype mandatory access control extensions to Xen [11]

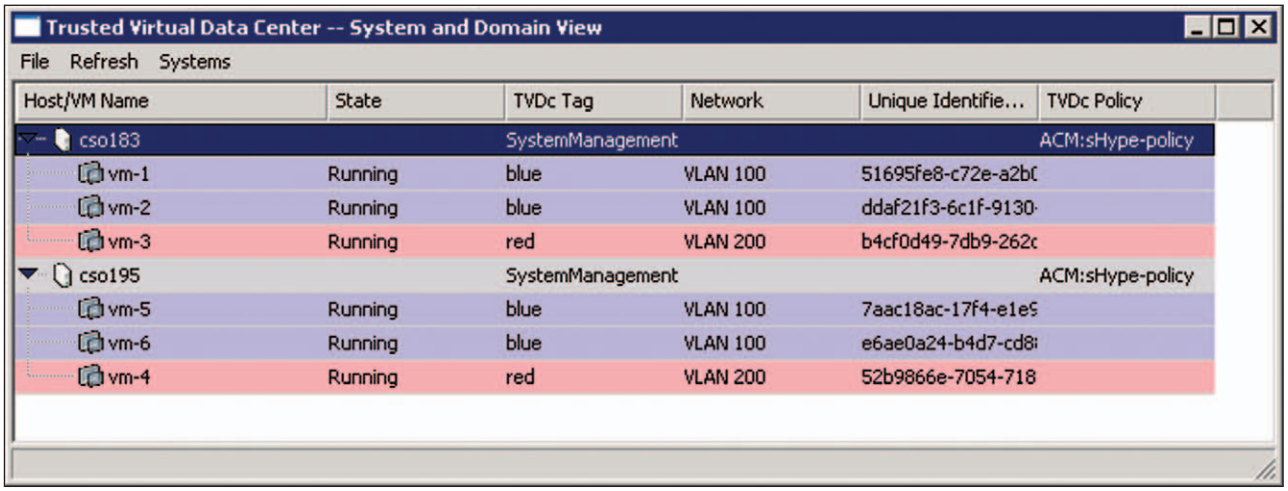

Figure 5

TVDc top management tool: System and domain view.

and the IBM PowerPC* [13] hypervisors. Previous implementations of hypervisor mandatory access control include the KVM/370 security kernel based on VM/370 [24] and the A1 secure VMM for VAX [25].

Work related to vTPM includes Sadeghi et al. [26] and Scarlata et al. [27]. Sadeghi et al. discuss the use of property-based attestation to address the fragility of platform configuration register (PCR) authorization due to software updates and VM migration. Scarlata et al. propose a generalized virtual TPM framework similar to ours to provide low-level TPM operations to applications running inside VMs.

Reference [21] describes a method that assigns VLANs to separate TVDs, which is similar to our network isolation approach.

References [28] and [29] describe an approach to strengthen grid security by the use of TPMs to provide cryptographic credentials, remote attestation, and integrity protection. This is similar to the distributed mandatory access control we use [30], which establishes verifiable trust in virtualized environments running a grid distributed application. Another ongoing work is the Open Trusted Computing [31] initiative, which proposes to develop a trusted and secure computing system based on open-source software.

NetTop [32] provides functionality similar to TVDc for client systems using commercially available hardware and software. NetTop uses virtualization to replace multiple end-user workstations having different security labels with VMs on a single physical system and uses virtual network configurations to interconnect VMs with secure enclaves. However, the NetTop architecture relies on the security controls of the host OS, since the VMM runs on top of the underlying OS. Our approach, in contrast, provides VM access control at the lowest levels of the system.

Conclusions and future work

Running workloads of different customers on the same machine increases security vulnerabilities and the potential for inadvertent or malicious loss of sensitive data and denial-of-service attacks. Furthermore, as long as cloud software stacks consist of a large number of heterogeneous components, there is an increased possibility of misconfiguration, which is an additional source of increased vulnerability. Therefore, isolation management, workload management, and access control are important aspects of cloud computing.

In this paper we extended previous work on TVDc, a technology developed to address the need for strong isolation and integrity guarantees in virtualized, cloud computing environments. TVDc uses a high-level specification of isolation and integrity requirements and constraints that support simplified, policy-driven security management, enabling quick and consistent response to dynamic data center conditions. The high-level policy drives the consistent configuration of distributed cloud resources such as servers and hypervisors, network infrastructure, and storage. TVDc combines coarse-grained isolation and trusted computing technologies to provide containment and trust properties across large, virtualized computing environments. Additionally, it builds on a split of management responsibilities between the cloud and tenant administrators.

A number of significant challenges remain for those whose aim is to achieve secure cloud computing. TVDc is only a first step in this direction.

We envision that the concepts of TVD and TVDc will serve as building blocks for future cloud computing architectures. Whereas it is apparent that the concepts of isolation and integrity management (embedded in TVD and TVDc as described above), together with centralized policy administration and distributed enforcement, enable one to simplify the management of the many resources in data centers, these concepts have to be developed further to provide end-to-end security guarantees. We anticipate that this can be done in three dimensions: (a) horizontally, to uniformly administer resources such as servers, storage, and networks; (b) vertically, to ensure that intra-tenant isolation is enforced first at the service or application level, and later at the lower levels of the infrastructure; and (c) from the tenant life-cycle perspective, to treat the concept of a tenant as a first-class object so it can be used as a basis for provisioning in the management plane as well as a meta-data artifact in the control plane. In particular, we foresee the following avenues for research:

Raising the discourse on isolation and integrity mechanisms and policies to the level of applications and services—Typically, a business customer's view is at the level of the applications and services that are deployed for it in the cloud or data center environment. In this sense, a customer is a tenant and in moving from an on-premises environment to a (possibly remote) hosting one, its primary concerns are about protecting the availability, integrity, and confidentiality of its business assets while ensuring that all operations are compliant with government and industry regulations. To ensure this, it is imperative that the tenant service-level agreement and external regulations be mapped to enforceable data center policies that can be integrated, managed, and enforced as part of the systems management infrastructure for the data center. In particular, since such policies are tenant-specific, we need to treat each tenant as a first-class object, attach policies to each tenant, and ensure that these policies are used in the provisioning, deployment, and management of the tenant TVD in the data center. We anticipate that such policies, phrased in terms of higher-level applications and services, will be refined to infrastructural level polices at lower levels, which can be managed and enforced as described above.

Implementing controlled sharing between TVDs—Whereas a TVD is a suitable abstraction for managing tenants in a data center environment, it is clear that a TVD cannot be a closed entity. In a manner similar to the interactions between consumers and businesses as well as between business partners, it is necessary to carefully relax the boundaries of a TVD to enable and promote such interactions where appropriate. We anticipate that one will enable and control both transactions as well as share information between TVDs as well as with external entities such as end users and business partners and suppliers by implementing filtering mechanisms, called *guards*. These guards would enforce policies on the flow of information and would be integrated into and managed by the centralized policy administration for the TVDc.

Acknowledgments

A number of colleagues have contributed the TVDc and to the related efforts on secure virtualization and trusted computing. We especially thank William Armstrong, Steven Bade, Leendert van Doorn, Peter Heyrman, Paul Karger, Stephen Levesque, José Moreira, Mike Williams, the POWER Hypervisor team, the System x* Virtualization team, and the Linux** Technology Center (LTC) Security team. We also thank the Xen community for their support.

References

1. S. Berger, R. Cáceres, D. Pendarakis, R. Sailer, E. Valdez, R. Perez, W. Schildhauer, and D. Srinivasan, "TVDc: Managing Security in the Trusted Virtual Datacenter," *Operating Systems Rev.* **42**, No. 1, 40–47 (2008).
2. A. Bussani, J. L. Griffin, B. Jasen, K. Julisch, G. Karjoth, H. Maruyama, M. Nakamura, et al., "Trusted Virtual Domains: Secure Foundations for Business and IT Services," *Research Report RC23792,* IBM Corporation (November 2005).
3. J. L. Griffin, T. Jaeger, R. Perez, R. Sailer, L. van Doorn, and R. Cáceres, "Trusted Virtual Domains: Toward Secure Distributed Services," *Proceedings of the First IEEE Workshop on Hot Topics in System Dependability*, Yokohama, Japan, June 30, 2005.
4. *IEEE Standard 802.1Q-2005*, "Virtual Bridged Local Area Networks," ©2005 IEEE; see *http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf*.
5. Trusted Computing Group; see *https://www.trustedcomputinggroup.org/home*.
6. W. E. Boebert and R. Y. Kain, "A Practical Alternative to Hierarchical Integrity Policies," *Proceedings of the 8th National Computer Security Conference*, 1985, pp. 18–27.
7. H. Maruyama, F. Seliger, N. Nagaratnam, T. Ebringer, S. Munetoh, S. Yoshihama, and T. Nakamura, "Trusted Platform on Demand," *Technical Report RT0564*, IBM Corporation (2004).
8. R. Sailer, X. Zhang, T. Jaeger, and L. van Doorn, "Design and Implementation of a TCG-Based Integrity Measurement Architecture," *Proceedings of the 13th USENIX Security Symposium*, August 9–13, 2004, San Diego, CA, pp. 223–238.
9. IBM Corporation, *IBM BladeCenter*; see *http://www-03.ibm.com/systems/bladecenter/*.
10. Cambridge University, UK,*The Xen Virtual Machine Monitor*; see *http://www.cl.cam.ac.uk/research/srg/netos/xen/*.

11. R. Sailer, T. Jaeger, E. Valdez, R. Cáceres, R. Perez, S. Berger, J. L. Griffin, and L. van Doorn, "Building a MAC-Based Security Architecture for the Xen Opensource Hypervisor," *Proceedings of the 21st Annual Computer Security Applications Conference (ACSAC 2005)*, Tucson, AZ, December 5–9, 2005, pp. 276–285.
12. J. P. Anderson, "Computer Security Technology Planning Study," *ESD-TR-73-51*, Vols. I–II, U.S. Air Force Electronic Division Systems, L. G. Hanscom Field, Bedford, MA, October 1972.
13. E. Valdez, R. Sailer, and R. Perez, "Retrofitting the IBM POWER Hypervisor to Support Mandatory Access Control," *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC)*, December 10–14, 2007, Miami Beach, Applied Computer Security Associates, pp. 221–231.
14. M. Factor, D. Naor, E. Rom, J. Satran, and S. Tal, "Capability Based Secure Access Control to Networked Storage Devices," *Proceedings of the 24th IEEE Conference on Mass Storage Systems and Technologies (MSST 2007)*, September 24–27, 2007, San Diego, CA, pp. 114–128.
15. S. Berger, R. Cáceres, K. A. Goldman, R. Perez, R. Sailer, and L. van Doorn, "vTPM: Virtualizing the Trusted Platform Module," *Proceedings of the 15th USENIX Security Symposium* **15**, No. 21, July 31–August 4, 2006, Vancouver, Canada, pp. 305–320.
16. The Trusted Computing Group, Trusted Platform Module (TPM) Specifications; see *https://www.trustedcomputinggroup.org/*.
17. IBM Corporation, IBM 4764 PCI-X Cryptographic Coprocessor; see *http://www-03.ibm.com/security/cryptocards/pcixcc/overview.shtml*.
18. Python Software Foundation, Python Programming Language; see *http://www.python.org*.
19. IBM Corporation, IBM Systems Director; see *http://www.ibm.com/systems/management/director/index.html*.
20. Intel Developer Forum, September 18–20, 2007, San Francisco, CA; see *http://www.intel.com/idf/us/fall2007/index.htm*.
21. S. Cabuk, C. I. Dalton, H. Ramasamy, and M. Schunter, "Towards Automated Provisioning of Secure Virtualized Networks," *Proceedings of the 14th ACM Conference on Computer and Communications Security*, October 29–November 2, 2007, Alexandria, VA, pp. 235–245.
22. Department of Defense, U.S. Government, *Department of Defense Trusted Computer System Evaluation Criteria (Orange Book)*, DoD 5200.28-STD, 1985.
23. T. Jaeger, R. Sailer, and U. Shankar, "PRIMA: Policy-Reduced Integrity Measurement Architecture," *Proceedings of the 11th ACM Symposium on Access Control Models and Technologies (SACMAT)*, June 7–9, 2006, Lake Tahoe, CA, pp. 19–28.
24. B. D. Gold, R. R. Linde, and P. F. Cudney, "KVM/370 in Retrospect," *Proceedings of the IEEE Symposium on Security and Privacy*, April 1984, Oakland, CA, pp. 13–23.
25. P. A. Karger, M. E. Zurko, D. W. Bonin, A. H. Mason, and C. E. Kahn, "A Retrospective on the VAX VMM Security Kernel," *IEEE Trans. Software Eng.* **17**, No. 11, 1147–1165 (1991).
26. A.-R. Sadeghi, C. Stüble, and M. Winandy, "Property-Based TPM Virtualization," *Proceedings of the 11th International Conference on Information Security*, September 15–18, 2008, Taipei, Taiwan, *Lecture Notes in Computer Science* 5222, Springer, pp. 1–16.
27. V. Scarlata, C. Rozas, M. Wiseman, D. Grawrock, and C. Vishik, "TPM Virtualization: Building a General Framework," *Trusted Computing*, N. Pohlmann and H. Reimer, Eds. Vieweg & Teubner, Wiesbaden, Germany, 2008, pp. 43–56.
28. W. Mao, H. Jin, and A. Martin, "Innovations for Grid Security from Trusted Computing," *Proceedings of International Symposium on Grid Computing*, April 29, 2005.
29. W. Mao, F. Yan, and C. Chen, "Daonity: Grid Security with Behavior Conformity from Trusted Computing," *Proceedings of the 1st ACM Workshop on Scalable Trusted Computing* (STC 2006), 2006.
30. J. M. McCune, T. Jaeger, S. Berger, R. Cáceres, and R. Sailer, "Shamon—A System for Distributed Mandatory Access Control," *Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC)*, Applied Computer Security Associates, 2006, pp. 23–32.
31. Open Trusted Computing; see *http://www.opentc.net*.
32. R. Meushaw and D. Simard, "NetTop-Commercial Technology in High Assurance Applications," *Tech Trend Notes*, National Security Agency, Fall 2000.

Received July 23, 2008; accepted for publication October 15, 2008

Stefan Berger *IBM Research Division, Thomas J. Watson Research Center, 19 Skyline Drive, Hawthorne, New York 10532 (stefanb@us.ibm.com)*. Mr. Berger, a Senior Software Engineer in the Secure Systems department, joined the IBM Research Division in 2000. His main interests are in virtualization, security, and operating systems. He is an active contributor to Xen hypervisor development and has been involved in the design and development of the virtualized Trusted Platform Module (TPM) and the sHype security architecture in Xen. He is the IBM representative to the Virtualized Platform Working Group in the Trusted Computing Group (TCG). He received his diploma in electrical engineering from the University of Erlangen–Nuremberg, Germany.

Ramón Cáceres *AT&T Labs, 180 Park Avenue, Florham Park, New Jersey 07932 (ramon@research.att.com)*. Dr. Cáceres is a Lead Member of Technical Staff at AT&T Labs. His research interests include mobile/pervasive/ubiquitous computing, wireless networking, virtualization, and security. Previous positions include Research Staff Member at the IBM Research Division and Chief Scientist at Vindigo, an award-winning provider of location-based services for mobile devices. He holds a Ph.D. degree from the University of California at Berkeley and is an ACM Distinguished Scientist.

Ken Goldman *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (kgold@watson.ibm.com)*. Mr. Goldman, a Senior Engineer in the Secure Systems group, has spent 10 years developing and implementing security solutions. He is currently the IBM representative to the TCG Trusted Computing Module Working Group, which develops the TPM standard. He implemented a software version of TPM suitable for virtualization, and is consulting throughout IBM on TCG technology. He received a B.S. degree in engineering physics from Cornell University and an M.S. degree in electrical engineering from MIT.

Dimitrios Pendarakis *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (dimitris@us.ibm.com)*. Dr. Pendarakis, a Research Staff Member and Manager of the Secure Systems group, has previously worked in various areas related to computer security, including IPsec/IKE, secure multi-party communication, and secure messaging with applications to sensor networks. His current research interests include secure virtualization and cloud computing, trusted computing, and secure hardware. He joined IBM in 1995 after receiving his Ph.D. degree from Columbia University. Between 2000 and 2003, he was Principal Architect at Tellium, where he led the development of next-generation control systems for intelligent optical networks.

Ronald Perez *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (ronpz@us.ibm.com)*. Mr. Perez, a Senior Technical Staff Member and Senior Manager, manages the Systems Solutions and Architecture department. In this role, he leads multiple teams of scientists and engineers pursuing advances in a diverse set of systems technologies, including virtualization and systems management, next-generation memory subsystems, multimedia, and information theory. He received his degree in computer science from University of Texas at Austin, and before joining IBM in 1997 he held positions with Motorola and Tandem Computers. He received an IBM Outstanding Technical Achievement Award in 2005 for his work on secure hypervisors and an IBM Outstanding Innovation Award in 2008 for his contributions to secure cryptographic coprocessors. He is Vice President in charge of the Trusted Computing Group open security standards organization and a senior member of the Association for Computing Machinery.

J. R. Rao *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (jrrao@us.ibm.com)*. Dr. Rao leads the Security department, with activities in Web services security, language-based security, security services, systems and network security, embedded systems security, hardware security, applied cryptography, and side-channel cryptanalysis. He has a Ph.D. degree in computer science from the University of Texas at Austin (1992), an M.S. degree in computer science from the State University of New York at Stony Brook (1986), and a B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur (1984). He is a member of the IFIP Working Group 2.3 (Programming Methodology) and is on the Steering Committee of the CHES Conference. He has published widely in a number of security conferences and holds a number of U.S. and European patents.

Eran Rom *IBM Research Division, Haifa Research Labs, Haifa University Campus, Mount Carmel, Haifa 31905, Israel (eranr@il.ibm.com)*. Mr. Rom received his M.S. degree in computer science from Tel-Aviv University in 2006. He subsequently joined the IBM Haifa Research Laboratory to work on storage security. Currently, he is involved in research in the area of distributed, scalable, and available storage systems.

Reiner Sailer *IBM Research Division, Thomas J. Watson Research Center, 19 Skyline Drive, Hawthorne, New York 10532 (sailer@us.ibm.com)*. Dr. Sailer, a Research Staff Member and Manager of the Security Services (GSAL) team, has previously led the Trusted Virtual Datacenter project. He received his diploma in computer science from Karlsruhe University, Germany, in 1994, and his Ph.D. degree in electronic engineering from the University of Stuttgart, Germany, in 1999. He subsequently joined the IBM Research Division to work on secure systems, trusted computing, virtualization security, and security services. At IBM, he has received several Outstanding Technical Achievement Awards in the areas of secure hardware and secure hypervisors.

Wayne Schildhauer *IBM Systems and Technology Group, P.O. Box 12195, Research Triangle Park, North Carolina 27709 (wschildh@us.ibm.com)*. Mr. Schildhauer is a Senior Software Engineer in the Blades and Modular Systems Virtualization group. He received a B.S.E. degree in electrical engineering from Duke University in 1979 and an M.S. degree in computer science from George Washington University in 1983. Since rejoining IBM at Tivoli Systems, his primary area of responsibility has been development of systems management software, with a recent emphasis on virtualization.

Deepa Srinivasan *IBM Systems and Technology Group, P.O. Box 12195, Research Triangle Park, North Carolina 27709 (deepas@us.ibm.com)*. Ms. Srinivasan is an Advisory Software Engineer in the Blades and Modular Systems Virtualization group. She received a B.S. degree in mathematics from the University of Madras, India, in 1997 and an M.S. degree in computer science from the Oregon Graduate Institute in 2003. Since joining IBM in 2001, she has worked on server consolidation tools development, BladeCenter management firmware development, and virtualization technologies.

Sivan Tal *IBM Research Division, Haifa Research Labs, Haifa University Campus, Mount Carmel, Haifa 31905, Israel (sivant@il.ibm.com)*. Mr. Tal is a Research Staff Member in the System Technologies and Services department. He received his B.A. degree in computer sciences from the Technion–Israel Institute of Technology in 1998. He subsequently joined the IBM Haifa Research Laboratory to work on advanced storage technologies. In recent years, he has led the development of the CbCS solution and the standardization of the protocol in the T10 technical committee of the International Committee for Information Technology Standards.

Enriquillo Valdez *IBM Research Division, Thomas J. Watson Research Center, 19 Skyline Drive, Hawthorne, New York 10532 (rvaldez@us.ibm.com)*. Dr. Valdez, a Senior Software Engineer, received his Ph.D. degree in computer science from Polytechnic University, Brooklyn, New York. He has led the design and implementation of the sHype security architecture in the IBM POWER Hypervisor* and has contributed to network isolation in the trusted virtual data center project.