Cloud Computing: Trust Management in Virtual Datacenters

Within the virtual architecture, there are some special considerations for securing your systems and establishing an organized datacenter.

Kai Hwang, Jack Dongarra, Geoffrey Fox

Adapted from "Distributed and Cloud Computing: From Parallel Processing to the Internet of Things" (Syngress, an imprint of Elsevier)

The architecture of virtual and cloud computing raises some unique concerns when it comes to security and access control. A virtual machine manager (VMM) changes the entire picture of computer architecture. It provides a layer of software between the OS and system hardware to create one or more virtual machines (VMs) on a single physical platform.

A VM entirely encapsulates the state of the guest OS it's hosting. You can copy and share an encapsulated machine state over the network and remove it like a normal file. This poses a whole layer of additional challenges to VM security.

In general, a VMM can provide secure isolation. A VM accesses hardware resources through the control of the VMM, so the VMM is the base of security for a virtual system. Normally, one VM is taken as a management VM to control privileges such as creating, suspending, resuming or deleting a VM.

Once a hacker successfully enters the VMM or management VM, the whole system is endangered. A more subtle problem arises in protocols that rely on the "freshness" of their random number source for generating session keys. Considering how a VM works, rolling back to a point after a random number has been chosen, but before it has been used, resumes execution. The random number, which must be "fresh" for security purposes, is then reused.

With a stream cipher, two different plain texts could be encrypted under the same key stream. This could, in turn, expose both plain texts if they have sufficient redundancy. Non-cryptographic protocols that rely on freshness are also at risk. For example, the reuse of TCP initial sequence numbers can raise the likelihood and severity of TCP hijacking attacks.

VM-Based Intrusion Detection

An intrusion is an unauthorized access to a certain computer from local or network users. Intrusion detection is used to recognize any unauthorized access. An intrusion detection system (IDS) is built on the OS, and is based on the characteristics of intrusion actions.

A typical IDS can be classified as a host-based IDS (HIDS) or a network-based IDS (NIDS), depending on the data source. You can implement an HIDS on the monitored system. When the monitored system is attacked by hackers, the HIDS also faces the risk of attack. An NIDS is based on the flow of network traffic that can't detect fake actions.

Virtualization-based intrusion detection can isolate guest VMs on the same hardware platform. Even some VMs are subject to successful invasion, but they never influence other VMs. This is similar to the manner in which an NIDS operates. Furthermore, a VMM monitors and audits access requests for hardware and system software, which can avoid fake actions. A VMM therefore has some similar qualities to an HIDS.

There are two different methods for implementing a VM-based IDS:

- The IDS is an independent process in each VM or a high-privileged VM on the VMM.

- The IDS is integrated into the VMM, and has the same hardware access privileges as the VMM.

The VM-based IDS contains a policy engine and a policy module. The policy framework can monitor events in different guest VMs through the OS interface library. PTrace indicates trace to secure policy of the monitored host. It's difficult to predict and prevent all intrusions without delay. Therefore, an analysis of the intrusion action following the intrusion is extremely important.

Most computer systems use logs to analyze attack actions, but it's hard to ensure a log's credibility and integrity. The IDS log service is based on the OS kernel. Thus, when an OS is invaded by attackers, the log service should be unaffected.

Besides using a full-blown IDS, you'll also find honeypots and honeynets commonly used in intrusion detection. They attract and provide a fake system view to attackers in order to protect the real system. You can also analyze the attack afterwards, and use that knowledge to build a more secure IDS.

A honeypot is a purposely defective system that simulates an OS to cheat and monitor an attacker's actions. You can divide a honeypot into physical and virtual forms. A guest OS and the applications running on it constitute a VM. The host OS and VMM must be guaranteed to prevent attacks from the VM in a virtual honeypot.

Trusted Zones

You can use industry solutions to build security middleware for trust management in distributed systems and private clouds. The concept of trusted zones was established as part of the virtual infrastructure (see **Figure 1**).

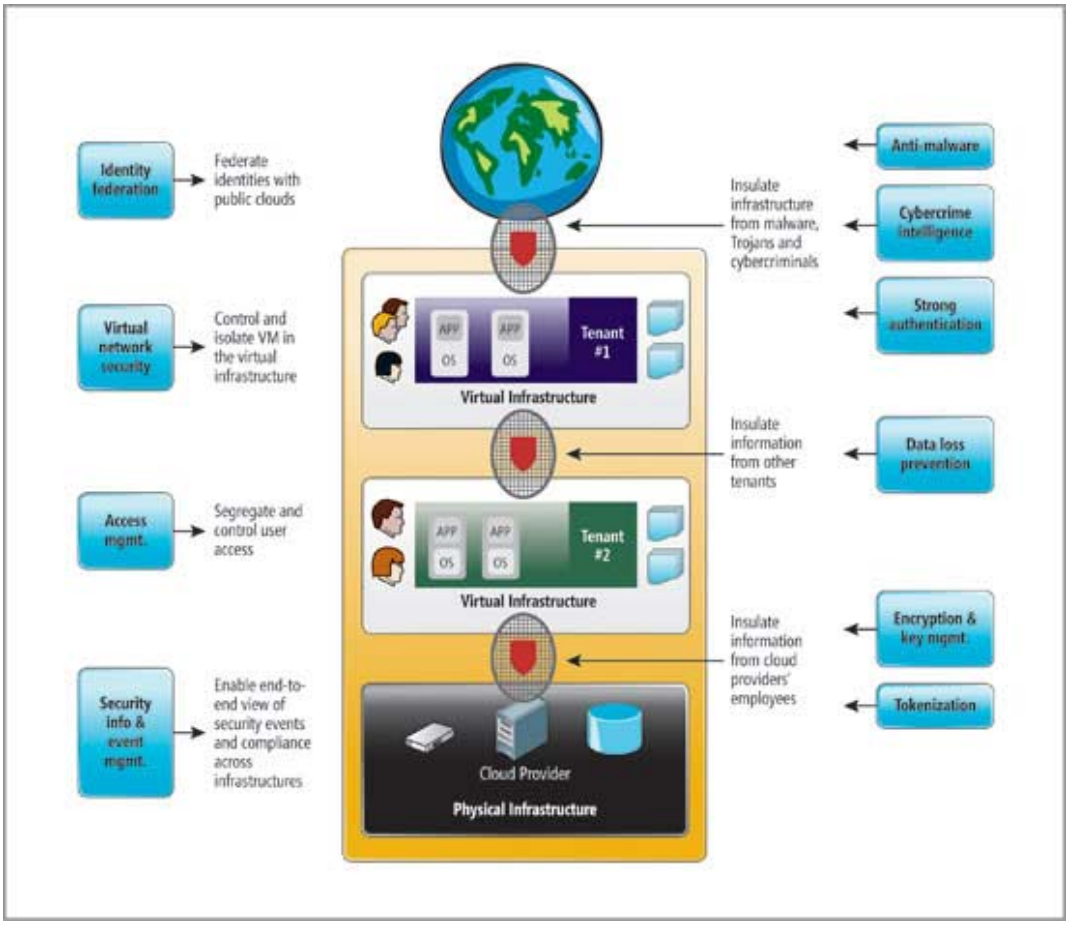

Figure 1 **The function and interaction of trusted zones.**

Create trusted zones for the virtual clusters (multiple applications and OSes for each tenant) provisioned in separate virtual

environments. The physical infrastructure is shown at the bottom, and is marked as a cloud provider. The virtual clusters or infrastructures are shown in the upper boxes for the two tenants. The public cloud is associated with the global user communities at the top.

The arrowed boxes on the left and the brief description between the arrows and the zoning boxes are security functions and actions taken at the four levels from the users to the providers. The small circles between the four boxes refer to interactions between users and providers and among the users themselves. The arrowed boxes on the right are functions and actions applied between the tenant environments, the provider and the global communities.

Almost all available countermeasures—antivirus, worm containment, intrusion detection, encryption and decryption mechanisms—are applied here to insulate the trusted zones and isolate VMs for private tenants.

The main innovation here is to establish the trust zones among the virtual clusters. The end result is an end-to-end view of security events and compliance across the virtual clusters dedicated to different tenants.

Kai Hwang *is a Professor of Computer Engineering for the University of Southern California and a visiting Chair Professor for Tsinghua University, China. He earned a Ph.D. in EECS from University of California at Berkeley. He has published extensively in computer architecture, digital arithmetic, parallel processing, distributed systems, Internet security, and cloud computing.*

Jack Dongarra *is a University Distinguished Professor of Electrical Engineering and Computer Science for the University of Tennessee, a Distinguished Research Staff at Oak Ridge National Laboratory and a Turning Fellow at the University of Manchester. Dongarra pioneered the areas of supercomputer benchmarks, numerical analysis, linear algebra solvers, and high-performance computing, and has published extensively in these areas.*

Geoffrey Fox *is a Distinguished Professor of Informatics, Computing and Physics and Associate Dean of Graduate studies and Research in the School of Informatics and Computing at Indiana University. He received his Ph.D. from Cambridge University, U.K. Fox is well known for his comprehensive work and extensive publications in parallel architecture, distributed programming, grid computing, web services, and Internet applications.*

Related Content

- Virtualization: Build an IT Lab for Virtual Machines
- Cloud Computing: Architecting a Microsoft Private Cloud
- Build a Green Datacenter