Security of the VMware vSphere® Hypervisor

UPDATED JANUARY 2014

Table of Contents

Executive Summary

Virtualization addresses IT's most pressing challenge: the infrastructure sprawl that compels IT departments to channel 70 percent of their budget into maintenance, leaving scant resources for business-building innovation.

The difficulty stems from the architecture of today's x86 computers: they're designed to run just one operating system (OS) and application at a time. As a result, even small data centers must deploy many servers, each operating at just 5–15 percent of capacity—highly inefficient by any standard.

Virtualization solves the problem by enabling several OSs and applications to run on one physical server or "host." Each self-contained "virtual machine," which comprises a guest OS and an application, is isolated from the others. It uses as much of the host's computing resources as it requires without affecting other virtual machines running on the same host.

VMware ESXi™ has been developed from the ground up to run virtual machines in a secure manner and incorporates many powerful security features that address the security concerns of the most demanding data center environments for enterprises and government organizations. The holistic security architecture of ESXi achieves this goal by providing security mechanisms at multiple layers:

- Secure isolation of virtual machines at the virtualization layer. This includes secure instruction isolation, memory isolation, device isolation, and managed resource usage and network isolation.
- Configurable secure management of the virtualized environment. This includes secure communication between virtualization components via SSL; host protection via lockdown mode; and least privilege by a fine-grained, role-based access-control mechanism.
- Secure deployment of the ESXi software on servers through use of various platform-integrity mechanisms such as digitally signed software packages and Intel Trusted Platform Module (TPM)–based trusted boot.
- Rigorous secure software development life cycle that enables developers to create software using secure design and coding principles such as minimum attack surface, least privilege, and defense in depth.

The success of this architecture in providing a secure virtualization and cloud infrastructure is evidenced by the fact that many large, security-conscious customers from areas such as banking and defense have chosen to trust their mission-critical services to VMware® virtualization. In fact, the most recent four versions of VMware vSphere® have been validated under the U.S. Common Criteria Evaluation and Validation Scheme (CCEVS) process, achieving EAL4+ certification.

In this document, we will elaborate on how ESXi security architecture and controls address common concerns in the security community regarding virtualization. This information is for experienced security professionals who are familiar with compute, storage, and networking technologies and with data center operations. Some of the concepts discussed in this paper might require a deeper understanding of general virtualization concepts. These are documented in the *Understanding Full Virtualization, Paravirtualization, and Hardware Assist and Virtualization Overview* white papers.

Secure Virtual Machine Isolation in Virtualization

Virtualization as it is done today has changed considerably since 2006. Then, it was done primarily in software using binary translation, a method that modifies sensitive instructions on the fly to "virtualizable" instructions. With advances in CPU technology, virtualization capabilities are built directly into the CPU.

Today, the hypervisor primarily is the management interface to the hardware primitives. Isolation of CPU, memory, and I/O now is done at a hardware level, with the hypervisor managing how much of the hardware resources a virtual machine can use, similar to a choreographer or traffic officer. This part of the hypervisor is called the virtual machine monitor (VMM). With the ability to leverage these CPU extensions, the attack surface of the hypervisor shrinks considerably.

Many security-related concerns about virtualization are unwarranted. Multiple hardware- and software-supported isolation techniques—as well as other robust security mechanisms such as access control and resource provisioning—address the risks associated with these worries. In the following sections, we will explain how these technologies address these concerns. For an in-depth view of the technologies used in VMware virtualization, refer to *Understanding Full Virtualization, Paravirtualization, and Hardware Assist*.

Virtualization Extensions

Intel VT-x and AMD-V both enable a VMM to give the CPU to a virtual machine for direct execution—an action called a virtual machine entry—until the time the virtual machine attempts to execute a privileged instruction. At that point, the virtual machine execution is suspended and the CPU is given back to the VMM—an action called a virtual machine exit. The VMM then follows the classic mainframe-era approach, inspecting the virtual machine instruction that caused the exit as well as other information provided by the hardware in response to the exit. With the relevant information collected, the VMM emulates the virtual machine privileged instruction against the virtual machine state and then resumes execution of the virtual machine with another virtual machine entry.

These technologies automatically trap sensitive events and instructions, eliminating the software overhead of monitoring all supervisory-level code for sensitive instructions. In this way, VT-x and AMD-V give the VMM the option of using either hardware-assisted virtualization or binary translation, depending on the workload. In some cases, workloads running on binary translation outperform hardware-assisted virtualization.

Instruction Isolation

From a security standpoint, a primary concern is that of a virtual machine's running in a highly privileged mode that enables it to compromise another virtual machine or the VMM itself. However, Intel VT-x and AMD-V extensions don't enable virtual machines to run at "Ring-0." Only the VMM runs at a hardware privilege level; guest OSs run at a virtualized privilege level. The guest OS does not detect that it is running at a nonprivileged virtualized level. As previously mentioned, when the guest OS executes a privileged instruction, it is trapped and emulated by the VMM.

Intel Hyper-Threading Technology (Intel HT Technology) enables two process threads to execute on the same CPU core. These threads can share the memory cache on the processor. ESXi virtual machines do not provide Intel HT Technology to the guest OS. ESXi, however, can utilize it to run two different virtual machines simultaneously on the same physical core if configured to do so.

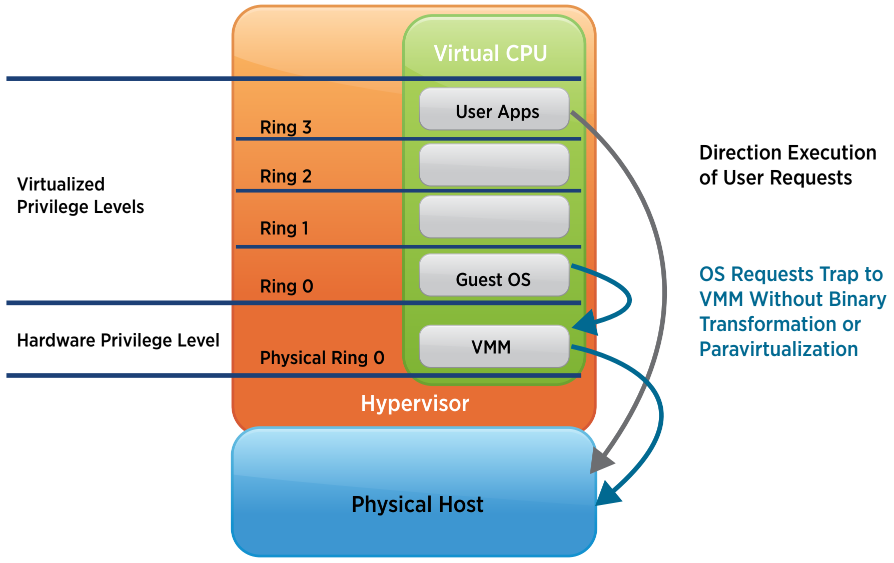


Figure 1. Instruction Isolation

Memory Isolation

The system administrator defines the RAM allocated to a virtual machine by the VMM via the virtual machine's settings. The VMkernel allocates memory when it defines the resources to be used by the virtual machine. A guest OS uses physical memory allocated to it by the VMkernel and defined in the virtual machine's configuration file.

An OS booting on real hardware is given a zero-based physical address space; an OS executing on virtual hardware is given a zero-based address space. The VMM gives each virtual machine the illusion that it is using such an address space, virtualizing physical memory by adding an extra level of address translation. A machine address refers to actual hardware memory; a physical address is a software abstraction used to provide the illusion of hardware memory to a virtual machine. This paper uses "physical" in quotation marks to distinguish this deviation from the usual meaning of the term.

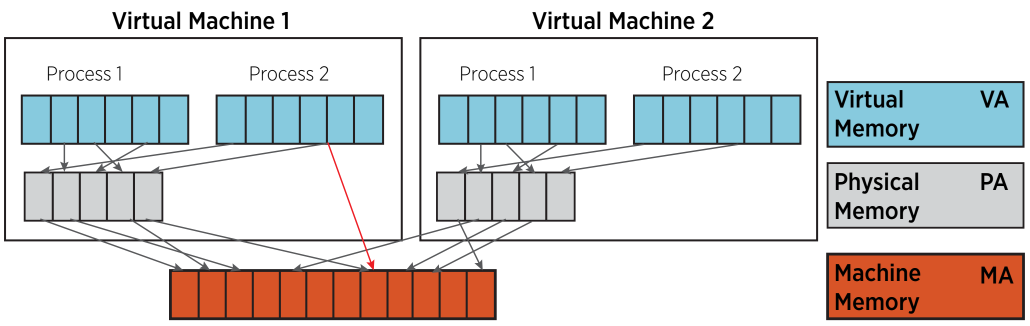


Figure 2. Memory Virtualization

The VMM maintains a pmap data structure for each virtual machine to translate “physical” page numbers (PPNs) to machine page numbers (MPNs). Virtual machine instructions that manipulate guest OS page tables or translation look-aside buffer contents are intercepted, preventing updates to the hardware memory management unit. Separate shadow page tables, which contain virtual-to-machine page mappings, are maintained for use by the processor and are kept consistent with the physical-to-machine mappings in the pmap.

This approach enables ordinary memory references to execute without additional overhead, because the hardware translation look-aside buffer caches direct virtual-to-machine address translations read from the shadow page table.

The extra level of indirection in the memory system is extremely powerful. The server can remap a “physical” page by changing its PPN-to-MPN mapping in a manner that is completely transparent to the virtual machine. It also enables the VMM to interpose on guest memory accesses. Any attempt by the OS or any application running inside a virtual machine to address memory outside of what has been allocated by the VMM causes a fault to be delivered to the guest OS, typically resulting in an immediate system crash, panic, or halt in the virtual machine, depending on the OS.

When a virtual machine requires memory, the VMkernel zeros each memory page out before being handed to the virtual machine. Normally, the virtual machine then has exclusive use of the memory page, and no other virtual machine can touch it or even detect it. The exception is when transparent page sharing is in effect.

Transparent page sharing is a technique for using memory resources more efficiently. ESXi scans the content of guest physical memory for sharing opportunities. Instead of comparing each byte of a candidate guest physical page to other pages, an action that is prohibitively expensive, ESXi uses hashing to identify potentially identical pages.

Memory pages that are identical in two or more virtual machines are stored once in the host system’s RAM, and each of the virtual machines has read-only access. Such shared pages are common, for example, if many virtual machines on the same host run the same OS. As soon as any one virtual machine attempts to modify a shared page, it gets its own private copy. Because shared memory pages are marked copy-on-write, it is impossible for one virtual machine to leak private information to another through this mechanism. Transparent page sharing is controlled by the VMkernel and VMM and cannot be compromised by virtual machines. It can also be disabled on a per-host or per–virtual machine basis.

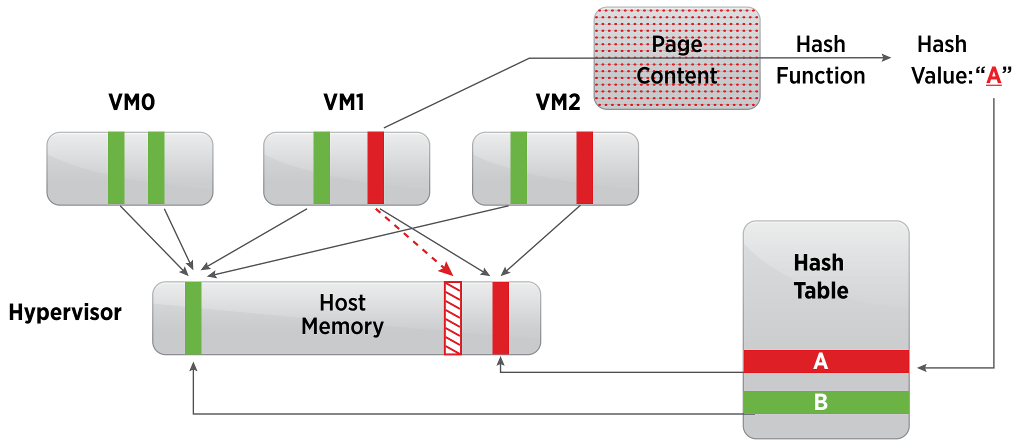


Figure 3. Transparent Page Sharing – Page-Content Hashing

Memory Protection

To protect privileged components, such as the VMM and VMkernel, vSphere uses certain well-known techniques. Address space layout randomization (ASLR) randomizes where core kernel modules are loaded into memory. The NX/XD CPU features enable the VMkernel to mark writeable areas of memory as nonexecutable. Both methods protect the system from buffer overflow attacks in running code. NX/XD CPU features also are exposed to guest virtual machines by default.

Device Isolation

Each virtual machine is isolated from other virtual machines running on the same hardware. Virtual machines share physical resources such as CPU, memory, and I/O devices; a guest OS in an individual virtual machine cannot detect any device other than the virtual devices made available to it.

To further clarify, a virtual machine can detect only the virtual (or physical) devices assigned to it by the systems administrator, such as the following examples:

- A virtual SCSI disk mapped to a file on a disk
- An actual disk or LUN connected to a physical host or array
- A virtual network controller connected to a virtual switch
- An actual network controller connected to a physical network

A virtual machine cannot map to a device that has not been preassigned. A virtual machine is incapable of mounting another virtual machine's disk unless the disk has been explicitly assigned to both virtual machines in the management console; for example, VMware vCenter™ or ESXi.

Device Access to Hardware

At the hardware level, all direct memory access (DMA) transfers and device-generated interrupts are virtualized and isolated from other virtual machines. This prevents one virtual machine from accessing the memory space controlled by another virtual machine. If such an attempt is made by a virtual machine, the guest OS will receive a fault from the CPU.

Because the VMkernel and VMM mediate access to the physical resources, and all physical hardware access takes place through the VMkernel, virtual machines cannot circumvent this level of isolation.

I/O Remapping

Modern processors feature an I/O memory management unit that remaps I/O DMA transfers and device interrupts. This enables virtual machines to have direct access to hardware I/O devices, such as network cards, storage controllers (HBAs), and GPUs. In AMD processors, this feature is called AMD I/O Virtualization (AMD-Vi) or I/O memory management unit (IOMMU); in Intel processors, the feature is called Intel Virtualization Technology for Directed I/O (VT-d). Within ESXi, use of this capability is called DirectPath I/O. DirectPath I/O does not impact the security properties in any way. For example, a virtual machine configured to use VT-d or AMD-Vi to directly access a device cannot influence or access the I/O of another virtual machine.

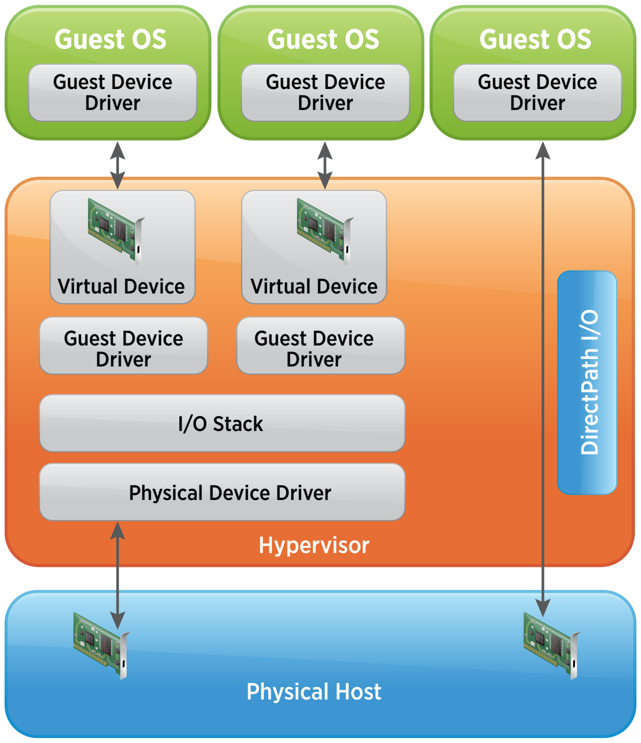


Figure 4. I/O Data Paths via the Hypervisor and DirectPath I/O

Resource Provisioning, Shares, and Limits

In a virtualized environment, resources are shared among all virtual machines. But because system resources can be managed, it enables use limits on virtual machines. There are a number of methods to address this.

Provisioning

In a physical system, the OS can use all the hardware resources. If the system has 128GB of memory, and the OS can address it, all of that memory can be used. The same applies to CPU resources. However, as previously noted, all resources are shared in a virtual environment. An OS using too many resources, CPU for example, potentially can deprive another OS of the resources it needs. Provisioning is the first step in managing virtual machine resources. A virtual machine should be provisioned with only the resources it requires to do a job. Because virtual machines never can use more CPU or memory resources than provisioned, users can limit the impact on other virtual machines.

Shares

Users can further isolate and protect neighboring virtual machines from “noisy neighbors” through the use of shares. Grouping “like” virtual machines into resource pools, and leaving shares set to default, ensures that all virtual machines in the pool receive approximately the same resource priority. A “noisy neighbor” will not be able to use more than any other virtual machine in the pool.

Limits

Previous recommendations suggested the use of limits to control resource usage. However, based on more operational experience, it has been found that virtual machine–level limits can have detrimental operational effects if used improperly.

For example, a virtual machine is provisioned with 4GB and the limit is set to 4GB. Then a change is made to increase the memory to 8GB, but the limit is left unchanged. At this stage, when the virtual machine has used up the 4GB and moves to an "extra 4GB," the 4GB limit causes the virtual machine to experience memory pressure and ballooning or swapping, thereby affecting its operational efficiency and the entire environment. Setting limits on a per–virtual machine basis also can have an impact on operational management and exposes the environment to misconfigurations. Limits are powerful but should be used only when the impact is fully understood.

For a deeper discussion of how to manage resource allocation for virtual machines, see the *vSphere Resource Management Guide*.

Network Isolation

ESXi Networks

There are a number of networks to consider on an ESXi server:

1. vSphere infrastructure networks, used for features such as VMware vSphere vMotion®, VMware vSphere Fault Tolerance, and storage. These networks are considered to be isolated for their specific functions and often are not routed outside a single physical set of server racks.
2. A management network that isolates client, command-line interface (CLI) or API, and third-party software traffic from normal traffic. This network should be accessible only by system, network, and security administrators. Use of "jump box" or virtual private network (VPN) to secure access to the management network is recommended. Access within this network to sources of malware should be strictly controlled.
3. Virtual machine networks can be one or many networks over which virtual machine traffic flows. Isolation of virtual machines within this network can be enhanced with the use of virtual firewall solutions that set firewall rules at the virtual network controller. These settings travel with the virtual machine as it migrates from host to host within a vSphere cluster.

See Figure 5 for a diagram of the increasing levels of sensitivity of the networks in a vSphere environment. The intent is to show that traffic for specific applications and functions should be isolated from one another. For example, vSphere vMotion traffic should not travel over networks where virtual machines are located. This prevents snooping of the traffic. Having separate networks also is recommended for performance reasons.

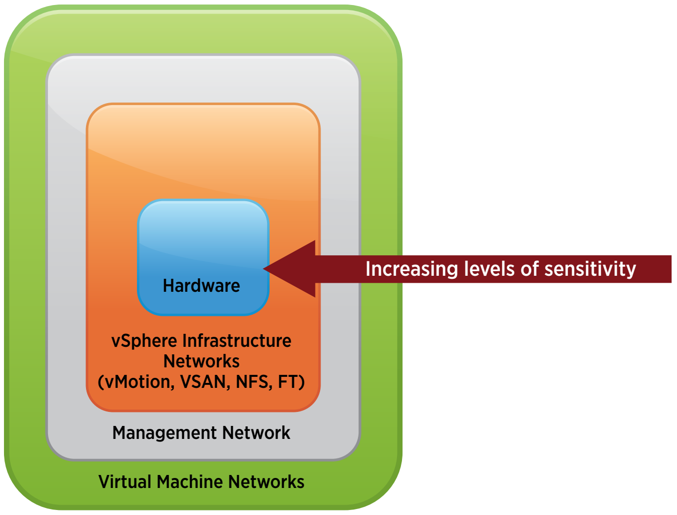

Hardware
Increasing levels of sensitivity
vSphere Infrastructure Networks (vMotion, VSAN, NFS, FT)
Management Network
Virtual Machine Networks

Figure 5. Increasing Sensitivity of Networks in Virtual Infrastructures

Virtual Machine Networks

Just as a physical machine can communicate with other machines in a network only through a network adapter, a virtual machine can communicate with other virtual machines running on the same ESXi host only through a virtual switch. Further, a virtual machine communicates with the physical network, including virtual machines on other ESXi hosts, only through a physical network adapter, unless it uses DirectPath I/O.

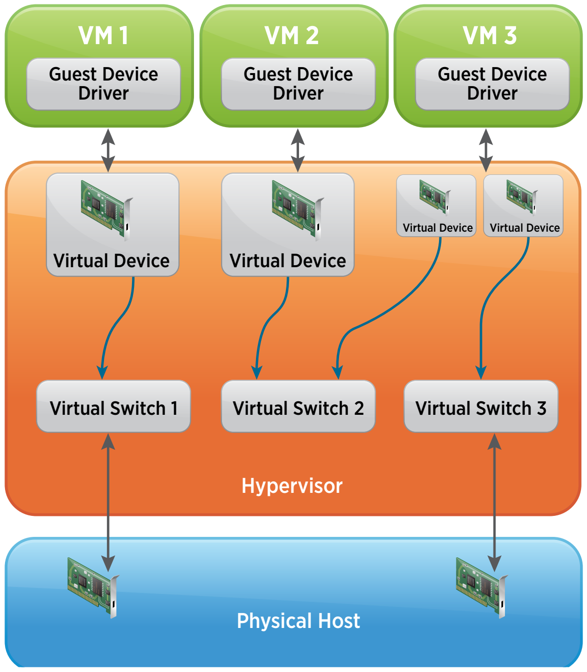


Figure 6. Network Isolation

In considering virtual machine isolation in a network context, users can apply these rules based on Figure 5:

- If a virtual machine does not share a virtual switch with any other virtual machine, it is completely isolated from other virtual networks within the host. This is virtual machine 1.
- If no physical network adapter is configured for a virtual machine, the virtual machine is completely isolated from any physical networks. This is virtual machine 2. In this example, the only access to a physical network is if virtual machine 3 acts as a router between virtual switch 2 and virtual switch 3.
- A virtual machine can span two or more virtual switches only if configured by the administrator. This is virtual machine 3.

Through the use of a virtualized network controller (vNIC)–level firewall, a virtual machine can be isolated from other virtual machines, even on the same switch (layer 2 isolation).

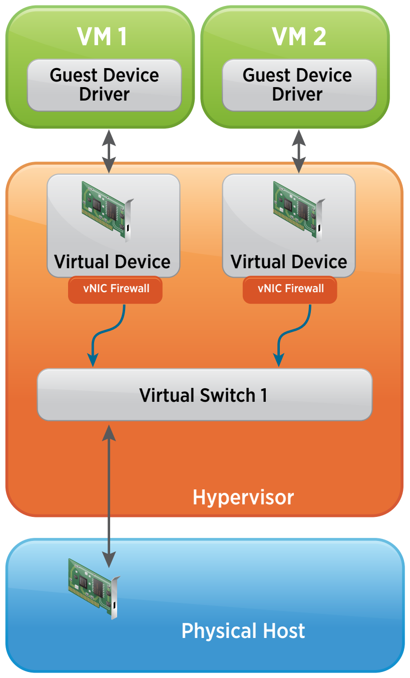


Figure 7. Virtual Firewall at the vNIC Level

These rules are applied to the vNIC of the virtual machine—not at the switch—enabling them to travel with the virtual machine.

Virtual Networking Layer

The virtual networking layer consists of the virtual network devices through which virtual machines interface with the rest of the network. ESXi relies on the virtual networking layer to support communication between virtual machines and their users. In addition, ESXi hosts use the virtual networking layer to communicate with iSCSI SANs, NAS storage, and so on. The virtual networking layer includes virtual network adapters and the virtual switches.

Virtual Switches

The networking stack uses a modular design for maximum flexibility. A virtual switch is "built to order" at runtime from a collection of small functional units, such as the following:

- The core layer 2 forwarding engine
- VLAN tagging, stripping, and filtering units
- Virtual port capabilities specific to a particular adapter or a particular port on a virtual switch
- Level security, checksum, and segmentation offload units

When the virtual switch is built at runtime, ESXi installs and runs only those components that are required to support the specific physical and virtual Ethernet adapter types used in the configuration. Therefore, the system pays the lowest possible cost in complexity and helps ensure a secure architecture.

Virtual Switch VLANs

ESXi supports IEEE 802.1q VLANs, which can be used to further protect the virtual machine network, management networks, and storage configuration. VMware software engineers wrote this driver in accordance with the IEEE specification. VLANs enable segmentation of a physical network so two machines on the same physical network cannot send packets to or receive packets from each other unless they are on the same VLAN.

There are three different configuration modes to tag—and untag—the packets for virtual machine frames:

- Virtual machine guest tagging (VGT mode) – Users can install an 802.1Q VLAN trunking driver inside the virtual machine. Tags will be preserved between the virtual machine networking stack and external switch when frames are passed from or to virtual switches.
- External switch tagging (EST mode) – Users can utilize external switches for VLAN tagging. This is similar to a physical network, and VLAN configuration normally is transparent to each individual physical server.
- Virtual switch tagging (VST mode) – In this mode, users provision one port group on a virtual switch for each VLAN; they then attach the virtual machine's virtual adapter to the port group instead of directly to the virtual switch. The virtual switch port group tags all outbound frames and removes tags for all inbound frames. It also ensures that frames on one VLAN do not leak into another VLAN. This is the most common type of configuration.

Virtual Ports

The virtual ports in ESXi provide a rich control channel for communication with the virtual Ethernet adapters attached to them. ESXi virtual ports authoritatively detect which are the configured receive filters for virtual Ethernet adapters attached to them, so no learning is required to populate forwarding tables.

They also authoritatively detect the "hard" configuration of the virtual Ethernet adapters attached to them. This capability makes it possible to set such policies as forbidding MAC address changes by the guest and rejecting forged MAC address transmission, because the virtual switch port can essentially authoritatively detect what is "burned into ROM"—actually, stored in the configuration file, outside the control of the guest OS.

The policies available in virtual ports are much more difficult—if not impossible—to implement with physical switches. Either ACLs must manually be programmed into the switch port, or weak conjecture such as "first MAC seen is assumed to be correct" must be relied on.

The port groups used in ESXi do not have a counterpart in physical networks. Think of them as templates for creating virtual ports with particular sets of specifications. Because virtual machines move from host to host, ESXi must have a good way to specify, through a layer of indirection, that a given virtual machine should have a particular type of connectivity on every host on which it might run. Port groups provide this layer of indirection, enabling VMware Infrastructure™ to provide consistent network access to a virtual machine, wherever it runs.

Port groups are user-named objects that contain enough of the following configuration information to provide persistent and consistent network access for virtual Ethernet adapters:

- Virtual switch name
- VLAN IDs and policies for tagging and filtering
- Teaming policy
- Layer security options
- Traffic-shaping parameters

Additional configuration options are available when using the VMware vSphere Distributed Switch™ (VDS), so port groups provide a powerful way to define and enforce security policies for virtual networking.

Virtual Network Adapters

vSphere provides several types of virtual network adapters that guest OSs can use. The choice of adapter depends upon factors such as support by the guest OS and performance, but all the adapters share the following characteristics:

- They have their own MAC addresses and unicast/multicast/broadcast filters.
- They are strictly layered Ethernet adapter devices.
- They interact with the low-level VMkernel layer stack via a common API.

Virtual Ethernet adapters connect to virtual ports when the user powers on the virtual machine on which the adapters are configured, takes an explicit action to connect the device, or migrates a virtual machine using vSphere vMotion. A virtual Ethernet adapter updates the virtual switch port with MAC filtering information when it is initialized and whenever it changes. A virtual port can ignore any requests from the virtual Ethernet adapter that would violate the layer 2 security policy in effect for the port.

Virtual Switch Isolation

Cascading—often required because physical switches have a limited number of ports—is a common cause of traffic leaks with physical switches. Because virtual switches provide all the required ports in one switch, there is no code to connect virtual switches. ESXi provides no path for network data to take between virtual switches, so it is relatively easy for ESXi to avoid accidental violations of network isolation or violations that result from a malicious user's or malicious software's running in a virtual machine.

In other words, the ESXi system does not have complicated and potentially failure-prone logic to ensure that only the right traffic travels from one virtual switch to another; instead, it simply does not implement any path that any traffic might use to travel between virtual switches. Furthermore, virtual switches cannot share physical Ethernet adapters, so it is not possible to manipulate the Ethernet adapter into doing loopback or something similar that would cause a leak between virtual switches.

In addition, each virtual switch has its own forwarding table, and there is no mechanism in the code to enable an entry in one table to point to a port on another virtual switch. In other words, every destination the switch looks up must match ports on the same virtual switch as the port where the frame originated, even if other virtual switches' lookup tables contain entries for that address. Because the VMkernel parses so little frame data—primarily only the Ethernet header—circumvention of virtual switch isolation would be difficult.

There are natural limits to this isolation. If a user connects the uplinks of two virtual switches, or bridges two virtual switches with software running in a virtual machine, it enables the same kinds of problems found with physical switches.

Virtual Switch Correctness

It is important to ensure that virtual machines or other nodes on the network not affect the performance of the virtual switch. ESXi guards against such influences in the following ways:

- Virtual switches do not gather information from the network to populate their forwarding tables. This eliminates a likely vector for denial-of-service (DoS) or leakage attacks, either as a direct DoS attempt or, more likely, as a side effect of some other attack such as a worm or virus as it scans for vulnerable hosts to infect.
- Virtual switches make private copies of any frame data used to make forwarding or filtering decisions. This is a critical feature and is unique to virtual switches.

It is essential to make certain that frames are contained within the appropriate VLAN on a virtual switch. ESXi does so in the following ways:

- VLAN data is carried outside the frame as it passes through the virtual switch. Filtering is a simple integer comparison. This is a special case of the general principle that the system not trust user-accessible data.
- Virtual switches have no dynamic trunking support.
- Virtual switches have no support for native VLAN.

Dynamic trunking and native VLAN are features through which an attacker might find vulnerabilities that might open isolation leaks. These features are not inherently insecure, but even when they are implemented securely, their complexity can lead to misconfiguration and open an attack vector.

For more information on virtual networking, see the *VMware Network Concepts* white paper.

Virtualized Storage

ESXi provides host-level storage virtualization, which logically abstracts the physical storage layer from virtual machines. An ESXi virtual machine uses a virtual disk to store its OS, program files, and other data associated with its activities. A virtual disk is a large physical file, or a set of files, that can be copied, moved, archived, and backed up as easily as any other file. Users can configure virtual machines with multiple virtual disks.

Each virtual disk resides on a datastore, such as VMware vSphere VMFS, a Network File System (NFS)–based datastore, or a virtual SAN datastore, which is deployed on physical storage. From the standpoint of the virtual machine, each virtual disk appears as if it were a SCSI drive connected to a SCSI controller. Whether the actual physical storage device is being accessed through local storage controller, iSCSI, NFS, Fibre Channel, or FCoE adapters on the host is transparent to the guest OS and to applications running on the virtual machine.

Each virtual disk—referred to as a `vmdk` file—is exclusively owned by a single powered-on virtual machine. No other virtual machine on the same or another ESXi host is allowed to access that virtual disk unless configured to do so; that is, Microsoft clusters using shared virtual SCSI disks. This situation does not change fundamentally when there is a cluster of ESXi hosts, with multiple virtual machines powered on and accessing virtual disks on a single VMware volume. Therefore, a vSphere vMotion operation, which enables live migration of a virtual machine from one ESXi host to another, is protected.

Linked Clones

A linked clone is made from a snapshot of the parent disk. All files available on the parent at the moment of the snapshot continue to remain available to the linked clone. Ongoing changes to the virtual disk of the parent do not affect the linked clone, and changes to the disk of the linked clone do not affect the parent.

NOTE: When linked clones of disks are used, the parent disk is shared but is in read-only mode. Writes are made to the snapshot or child disk. A clone cannot write to the parent disk.

Raw Device Mapping

In addition to virtual disks, vSphere offers a mechanism called raw device mapping (RDM). RDM is useful when a guest OS inside a virtual machine requires direct access to a storage device. Certain operational considerations and limitations exist when using RDMs. These are called out in VMware storage documentation.

Virtual machines can be configured to throttle the bandwidth they use to communicate with storage devices. This prevents the possibility of a denial-of-service attack against other virtual machines on the same host by one virtual machine attempting to use too many I/O operations on a shared storage device. It is configured using shares; the topic is discussed further in the "Storage I/O Control Resource Shares and Limits" subsection of the *vSphere Resource Management Guide*.

I/O Path

ESXi implements a streamlined path to provide high-speed and isolated I/O for performance-critical network and disk devices. An I/O request issued by a guest OS first goes to the appropriate driver in the virtual machine. To access a virtual disk, a virtual machine uses a virtual storage controller. ESXi emulates the following SCSI controllers in the guest:

- SATA
- SAS
- VMware paravirtualized
- LSI Logic
- BusLogic SCSI devices

The corresponding driver is loaded into the guest OS and typically turns the I/O requests into accesses to I/O ports to communicate to the virtual devices using privileged IA-32 IN and OUT instructions. These instructions are trapped by the VMM and then handled by device emulation code in the VMM, based on the specific I/O port being accessed. The VMM then calls device-independent network or disk code to process the I/O. For disk I/O, ESXi maintains a queue of pending requests per virtual machine for each target SCSI device. The disk I/O requests for a single target are processed in round-robin fashion across virtual machines by default. The I/O requests are then sent down to the device driver loaded into ESXi for the specific device on the physical machine.

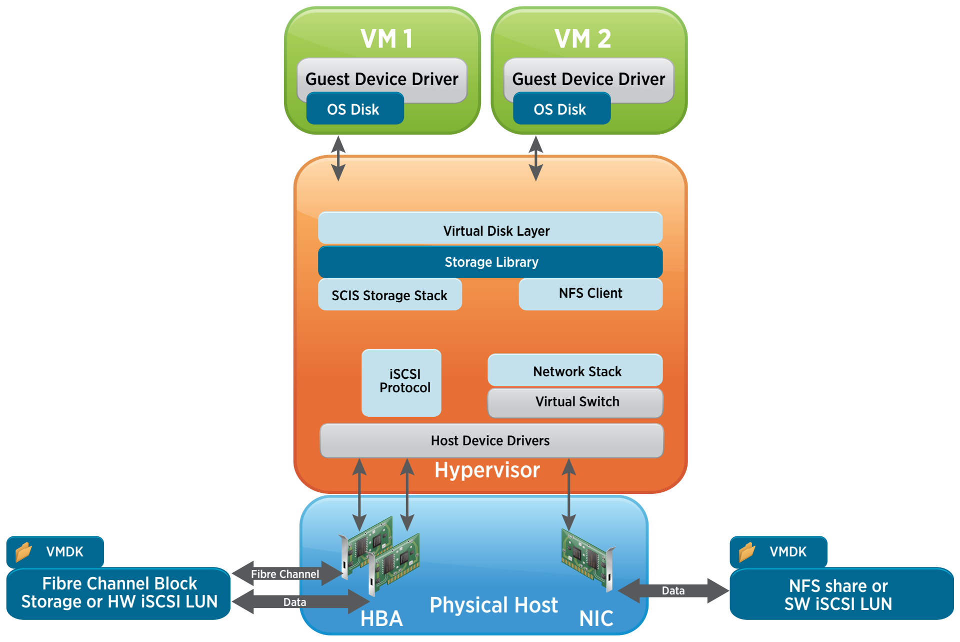


Figure 8. NFS and Block Storage I/O

SAN Security

A host running ESXi is attached to a Fibre Channel SAN in the same way as is any other host. It uses Fibre Channel HBAs, with the drivers for those HBAs installed in the software layer that interacts directly with the hardware. In environments that do not include virtualization software, the drivers are installed on the OS; with ESXi, the drivers are installed in the ESXi VMkernel.

iSCSI Security

ESXi supports unidirectional CHAP for all types of iSCSI initiators; it supports bidirectional CHAP for software and dependent hardware iSCSI.

VMFS

ESXi also includes VMFS, a distributed file system and volume manager that creates and manages virtual volumes on top of the LUNs that are presented to the ESXi host. These virtual volumes, usually referred to as virtual disks, are allocated to specific virtual machines. VMFS is used on SAN-based block storage and iSCSI network connections.

A VMware virtual machine has no visibility into the World Wide Name (WWN), the physical Fibre Channel HBAs, or the target ID or other information about the LUNs upon which its virtual disks reside. The virtual machine is isolated to such an extent that software executing in the virtual machine cannot detect that it is running on a SAN fabric, NAS, or host-based directly attached storage. Even multipathing is handled in a way that is transparent to a virtual machine.

Consider the example of running a Microsoft Windows OS inside a VMware virtual machine. The virtual machine detects only certain virtual disks: those chosen by the ESXi administrator at the time the virtual machine is configured. Such an operation is effectively LUN masking in the virtualized environment. It has the same security benefits as LUN masking in the physical world but is done with a different set of tools. Software executing in the virtual machine, including the Windows OS, detects only the virtual disks attached to the virtual machine. Even if the Windows OS attempts to issue a SCSI command—REPORT LUNS, for example—in an effort to discover other targets, ESXi prevents it from discovering any SCSI information that is not appropriate to its isolated and virtualized view of its storage environment.

Additional complexities in the storage environment arise when a cluster of ESXi hosts is accessing common targets or LUNs. VMFS ensures that all of the hosts in the cluster cooperate to ensure correct permissions and safe access to the VMware volumes. File locks are stored on disk as part of the volume metadata, and all ESXi hosts utilizing the volumes detect the ownership. Ownership of files and various distributed file system activities are rendered exclusive and atomic by the use of standard SCSI reservation primitives.

NFS Security

For NFS connections, ownership is root. The read/write no_root_squash sync settings are enabled for root to create or delete files on the NFS share. NFS locking on ESXi does not use the NLM protocol. VMware has established its own locking protocol. These NFS locks are implemented by creating lock files on the NFS server. To inform ESXi hosts that a lock is still active, VMware periodically sends updates to the lock file after it has been created. The lock file updates generate small, 84-byte WRITE requests to the NFS server.

Secure Management

Any IT infrastructure software must have a means for management, configuration, monitoring, and troubleshooting. ESXi has been designed to provide a rich set of management features that do not compromise the ability to maintain the security of the platform. First we will discuss the design of the ESXi hypervisor; then we will discuss capabilities related to access controls—such as authentication, authorization, and logging—as well as secure communications.

ESXi Is Not Linux

ESXi is a purpose-built system kernel designed from the ground up to run virtual machines and associated services. As such, it follows many POSIX constructs and semantics, but it is in many ways very different from a UNIX or Linux OS. Typical Linux concepts such as file systems and users and groups are not applicable to an ESXi system. The files that one sees when logged in to the shell exist only in memory, and changes made to most files are not persistent across reboots. ESXi also does not have a concept of “users” for the purpose of file or process ownership. All logins to the shell have exactly the same privileges, and every file is equally accessible to all shell sessions. Command auditing is the only area in which a username is a factor; every command that is executed is logged as being performed by the user who initiated that shell session. There also are no general-purpose services or processes running on the kernel. Other than the processes that support running virtual machines, the only services are those related to monitoring, management, and hardware management; for example, NTP, syslog, and SNMP.

The implications of this design are significant: Security hardening for a typical Linux system potentially requires dozens of steps, such as changing file permissions, disabling services, and managing user privileges; *ESXi* requires none of these steps, most of which either are impossible to perform or have no effect. The primary measures required for hardening an ESXi host involve securing and properly managing the interfaces with the system that are used for configuration, management, and troubleshooting.

Because ESXi is a single-user system on which all users run at the same privilege level as root, changing file permissions using CHMOD does not restrict user access to files and is not supported.

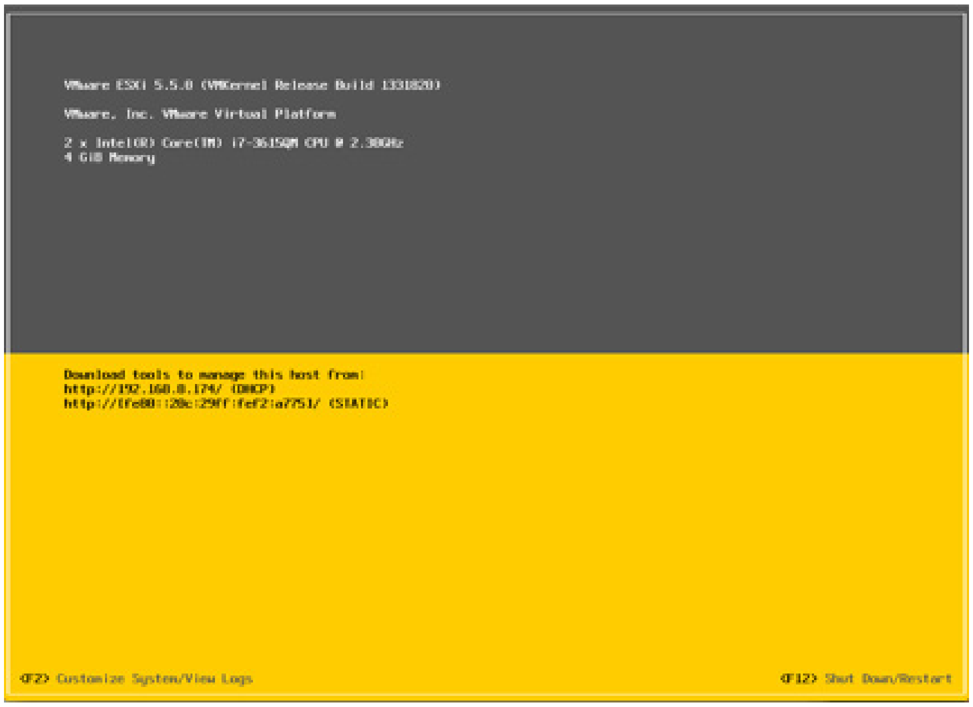
Disabling services, other than through a supported user interface or by procedures cited in vSphere documentation or a VMware knowledge base article, is not supported and can have an adverse effect on an ESXi system.

Administrative Interfaces

As part of the base ESXi image, there are three primary interfaces for administration.

DCUI

The direct console user interface (DCUI) is the primary one that is presented to the user when ESXi is installed for the first time. It is a menu-driven, text-based interface that appears on the physical console of the ESXi server hardware—that is, the screen that is plugged into the hardware graphics port—or via the remote console feature of some hosts; for example, HP Integrity Integrated Lights-Out (iLO) and Dell Remote Assistant Controller (DRAC). Through this interface, an administrator can perform basic tasks such as configuring the management network, resetting the root password, and rebooting the host. Any task that is more sophisticated requires using one of the other interfaces. After the host has been joined to VMware vCenter Server™, the administrator has the option of disabling the DCUI, forcing all host-level management interaction to occur through one of the other interfaces.



```
VMware ESXi 5.5.0 (VMKernel Release Build 1331820)

VMware, Inc. VMware Virtual Platform

2 x Intel(R) Core(TM) i7-3615QM CPU @ 2.30GHz
4 GiB Memory

Download tools to manage this host from:
http://192.168.0.174/ (DHCP)
http://[fe80::20c:29ff:fef2:a775]/ (STATIC)

<F2> Customize System/View Logs                    <F12> Shut Down/Restart
```

Figure 9. DCUI

Host Agent

The host agent, or *hostd*, is a process running on all ESXi hosts; it provides access to the host that is governed according to the VMware vSphere API model. It plays the role of converting API calls into the corresponding low-level actions on the ESXi host. The most popular method of utilizing this API is via CLIs. The two primary CLIs in vSphere are VMware vSphere PowerCLI™, based on Microsoft Windows PowerShell, and the vSphere CLI, a unique set of CLI tools that can be installed on any Linux or Windows system. Any software program that utilizes Web services can also communicate directly with the host agent, enabling more programmatic access. There are several language bindings available for the vSphere Web services API, including Java and .NET. The types of tasks that the host agent can process are far more complex than those possible with the DCUI.

Another tool that uses the host API model is VMware vSphere Client™, the legacy Windows-based graphical interface. vSphere Client is limited to vSphere 5.0 functionality and should be used only for standalone host operations and VMware vSphere Update Manager™ functionality.

VMware vSphere ESXi Shell

VMware vSphere ESXi Shell is an interactive CLI that can be used to perform monitoring and diagnostics. The most important element in the vSphere ESXi Shell is called ESXCLI, a command set that provides a full set of monitoring and configuration capabilities for nearly all aspects of ESXi functionality. With ESXCLI, an administrator can perform activities such as listing all network interfaces, performing low-level configuration of an HBA interface, querying kernel parameters, and so on.

Of the three interfaces, vSphere ESXi Shell is the most in-depth and complete, so it should be used only for troubleshooting and resolving problems. Unlike a typical Linux shell, vSphere ESXi Shell has a limited set of generic commands, such as for file or process manipulation. Access to vSphere ESXi Shell comes in two forms.

Local access is available on the local console of the host, as it is for the DCUI. A special key combination switches the display between DCUI and a login screen at which an authorized user must provide a username and password to initiate a vSphere ESXi Shell session.

Remote access is available to the host via SSH over the management network. An SSH connection also presents a username and password prompt to control a remote vSphere ESXi Shell session. Each of these forms of access can be disabled independently as part of the host's configuration. For example, the remote (SSH) access can be disabled while still allowing local console access.

In all cases, to be granted vSphere ESXi Shell access, an administrator role must be assigned to the user.

NOTE: All vSphere ESXi Shell commands are audited and logged to syslog.

Network Access to Administrative Interfaces

The administrative interfaces of vSphere can be compared to a physical data center, where limited access—via badges, for example—is a common security practice. Similarly, tightly controlled and monitored security techniques should be applied to accessing a virtual data center. Putting the management interfaces of ESXi and vCenter on a separate, access-controlled management LAN is recommended. Other solutions that require access to vCenter, such as backup software, should also be on this management LAN.

Each of the three administrative interfaces of ESXi has some form of network-based access; each one also has mechanisms to prevent unauthorized or unwanted access. The DCUI is available only on the console of the host, so the only way to access it over the network is by configuring the server's remote console option—HP iLO or Dell DRAC, for example. Therefore, access to the DCUI over the network is governed entirely by the controls put in place around the remote console.

vSphere ESXi Shell can be accessed via the local console of the host, so controls that govern remote access to this console pertain to vSphere ESXi Shell access as well. Unlike the DCUI, vSphere ESXi Shell can also be accessed remotely via SSH. In a typical SSH session, the host's SSH fingerprint presents itself the first time a client connects to the host. This verifies the identity of the host and prevents a malicious entity from masquerading as the host. Subsequently, a set of SSH keys that mark the host as trusted is created. It is possible, however, to previously upload a set of authorized SSH keys to the host. Because the identity of the client already has been verified by the presence of these authorized keys, trusted clients are enabled to connect to the host without having to check its fingerprint.

After a user has logged into vSphere ESXi Shell via SSH, they can issue the "dcui" command, which then displays a DCUI emulator in the vSphere ESXi Shell session. Because the vSphere ESXi Shell by itself enables much more sophisticated capabilities via the ESXCLI command set, this is provided as a convenience for users who are more familiar with the DCUI interface.

The host agent can be reached over the management network interface that has been configured for the ESXi host. This interface is completely different from those used by any guest virtual machines running on the host. In nearly all production environments, the management interface is on a VLAN or network that is separate from any guest network.

SSL Certificates

Network communication to the management interface is encrypted using SSL certificates. A host has self-signed certificates when first deployed, but these can be replaced by certificate authority (CA)–signed certificates if required by local policy. Self-signed certificates can be as secure as certificates that are issued by an external CA if the user validates the certificate and its SSL thumbprint when the warning dialog appears. The client application connecting to the ESXi host management interface determines whether or not a certificate has been properly signed, regardless of whether access is via vSphere Client, CLI, or API.

Management Interface Firewall

There is a firewall built into ESXi that can be used to restrict which network addresses are allowed to connect to the management interface. This firewall also can be used to restrict all communication to the ESXi host over the management interface, based on TCP port number. This enables functionality, other than administration, that operates based on network communication to the ESXi host. It spans the spectrum of vSphere virtualization features such as vSphere vMotion, VMware vSphere High Availability, VMware vSphere Fault Tolerance, IP-based storage, and so on, as well as other basic functions such as DNS, logging, NTP, and others.

Security of Management Protocols

vSphere employs a secure version of management protocols as part of its implementation. The ESXi SNMP agent can use SNMP v3. This provides stronger security than v1 or v2c, including key authentication and encryption. ESXI also supports UDP (default), TCP, and SSL for syslog message forwarding.

Administrative User Access

There are two primary ways in which to define and authenticate users who are able to log in to vSphere ESXi Shell and the other management interfaces of the host. The first way is to create local user accounts on the ESXi host directly. These users, and their passwords, are not synchronized with any other host, so each account creation must be done independently of one another. The second way is to use a centralized authentication directory, such as Microsoft Active Directory. When an ESXi host is joined to an Active Directory domain, it can check usernames and passwords against this central store without requiring any information to be stored locally. In addition, only users who are members of a specially designated Active Directory group are given permission to log in to vSphere ESXi Shell and DCUI. All other users will be rejected.

The only user who is predefined on ESXi is the *root* user. The root user has full administrative privileges. Administrators can use this login and its associated password to log in to a host. Root users have a complete range of control activities on the specific host they are logged in to, including manipulating permissions, creating local groups and users, working with events, and so on.

The root password is one of the parameters that must be provided as part of the installation process. The root account provides the means by which an initial connection can be made with the host. However, after this has been done—the host has been joined with vCenter Server, for example—it is often preferable that the root account not be used anymore. Because root is a generic account, anyone with knowledge of the root password can perform administrative tasks. But these activities cannot be unambiguously associated with the person performing them. Instead, all administrators should use an account personally associated with them (named account) to perform any management tasks, whether local accounts or ones based in Active Directory.

Limit Root Access

Because many processes run under the root account on ESXi, the root account on an ESXi server cannot be removed. Some local security policies require a user to be associated with every account. This presents a problem because in many cases it would require the removal of any account that does not have a user associated with it. In highly secure environments where root access must be severely limited and the ESXi system is connected to vCenter Server, the use of lockdown mode is the recommended procedure. For standalone ESXi servers, the use of a named account and the vaulting of the root account are recommended.

Lockdown Mode

Lockdown mode is a feature of vSphere that disables login and API functions from being executed directly on an ESXi server. It is available only on ESXi hosts that have been added to vCenter Server. When a host is in lockdown mode, users cannot run vSphere CLI commands from an administration server or from a script. Use of lockdown mode removes all vSphere API privileges associated with root. All operations must be performed via vCenter Server, with the following one exception.

The *vpxuser* user is a vCenter Server entity with root rights on the VMware ESX®/ESXi host, enabling it to manage activities for that host. Prior to vSphere 5.5, the vpxuser was created at the time an ESX/ESXi host was attached to vCenter Server. It was not present on the ESX host unless the host was being managed through vCenter Server. As of vSphere 5.5, the vpxuser account is present in ESXi hosts, regardless of whether they are connected to vCenter. In lockdown mode, only the vpxuser user has authentication permissions; no other users can perform operations against the host directly. The root account cannot be logged in to nor can API operations be run using it.

The DCUI is the one exception to this limiting of local operations. Users can be assigned DCUI access privileges explicitly via the DCUI access advanced configuration option, which has DCUI.Access as the key and a comma-separated list of ESXi users as the value. Users in the list can access the DCUI at any time, even if they are not administrators (admin role) and even when the host is in lockdown mode.

Enabling or disabling lockdown mode affects which types of users are authorized to access host services, but it does not affect the availability of those services. In other words, if vSphere ESXi Shell, SSH, or DCUI services are enabled, they will continue to run whether or not the host is in lockdown mode. Using vSphere Web Client to manage a host, or using the DCUI, users can enable lockdown mode using the Add Host wizard to add a host to vCenter Server.

Platform Integrity Protection

Secure Software Packaging

A vSphere installation bundle (VIB) is an ESXi software package. VMware and its partners package solutions, drivers, CIM providers, and applications that extend the ESXi platform; they are packaged as VIBs. VIBs generally are made available in software depots. Users can utilize VIBs to create and customize ESXi ISO images or to upgrade ESXi hosts by installing the VIBs asynchronously onto the hosts.

Each VIB is released with a rating called an "acceptance level." The host acceptance level determines which VIBs can be installed to a host. A digital signature guarantees the acceptance level of a VIB. VMware supports the following acceptance levels:

VMwareCertified	This acceptance level has the most stringent requirements. VIBs with this level go through thorough testing fully equivalent to VMware in-house quality assurance testing for the same technology. Other than software written by VMware, only IOVP drivers are published at this level. VMware takes support calls for VIBs with this acceptance level.
VMwareAccepted	VIBs with this acceptance level go through verification testing, but the tests do not cover every function of the software. A partner runs the tests and VMware verifies the results. Today, CIM providers and PSA plug-ins are among the VIBs published at this level. VMware directs support calls for VIBs with this acceptance level to the partner's support organization.
PartnerSupported	VIBs with this acceptance level are published by a partner that is trusted by VMware. The partner performs all testing. VMware does not verify the results. This level is used for a new or nonmainstream technology that partners want to enable for VMware systems. Today, driver VIB technologies such as InfiniBand, ATAoE, and SSD are at this level with nonstandard hardware drivers. VMware directs support calls for VIBs with this acceptance level to the partner's support organization.
CommunitySupported	This acceptance level is for VIBs created by individuals or companies outside of VMware partner programs. VIBs at this level have not gone through any testing program approved by VMware and are not supported by VMware technical support or by a VMware partner.

All ESXi binaries—kernel modules, applications, shared libraries—are packaged in a VIB. VMware signs each of these VIBs with the VMwareCertified acceptance level. VIBs with VMwareAccepted and PartnerSupported acceptance levels are signed by trusted VMware partners. CommunitySupported VIBs are not signed. PartnerSupported is the default acceptance level for ESXi 5.5.

IT administrators can create their own VIBs. For example, users can create a VIB that adds a firewall rule or a custom script to their ESXi system. They can incorporate that VIB into a custom ISO of ESXi if they have a local requirement for common settings at installation time. Because self-made VIBs cannot be signed, the caveat is that the acceptance level on the ESXi server must be reduced to CommunitySupported.

The following are examples of ESXCLI commands for managing VIBs and verifying acceptance levels:

```
# esxcli software vib list
Name                 Version                      Vendor Acceptance Level Install Date
-----------------    ---------------------------  ------ ---------------- ------------
ata-pata-amd         0.3.10-3vmw.500.0.0.469512   VMware VMwareCertified  2012-05-04
ata-pata-atiixp      0.4.6-3vmw.500.0.0.469512    VMware VMwareCertified  2012-05-04
ata-pata-cmd64x      0.2.5-3vmw.500.0.0.469512    VMware VMwareCertified  2012-05-04
ata-pata-hpt3x2n     0.3.4-3vmw.500.0.0.469512    VMware VMwareCertified  2012-05-04
```

Software Assurance and Integrity Protection

vSphere uses Intel Trusted Platform Module/Trusted Execution Technology (TPM/TXT) to provide remote attestation of the hypervisor image, based on hardware root of trust. The hypervisor image consists of the following elements:

- ESXi software (hypervisor) in VIB (package) format
- Third-party VIBs
- Third-party drivers

To leverage this capability, an ESXi system must have TPM and TXT enabled.

When TPM and TXT are enabled, ESXi measures the entire hypervisor stack when the system boots; it stores these measurements in the platform configuration registers (PCRs) of the TPM. The measurements include the VMkernel, kernel modules, drivers, native management applications that run on ESXi, and any boot-time configuration options. All VIBs that are installed on the system are measured. The measurements are exposed in a vSphere API. An event log is provided as part of the API, as specified by the Trusted Computing Group (TCG) standard for TXT.

By comparing this image to an image of the expected known good values, third-party solutions can leverage this feature to detect tampering of the hypervisor image.

NOTE: Users must utilize a third-party solution to fully leverage vSphere TPM/TXT baseline support for hypervisor attestation. vSphere does not provide a user interface to view these measurements.

VMware Secure Development Life Cycle

To improve upon the security of its products, VMware uses a number of techniques during its software development cycle. These standard techniques, using both internal and external security expertise, include threat modeling, static code analysis, incident response planning, and penetration testing.

VMware has an established Software Security Engineering group that integrates these techniques into the software development cycle and provides security expertise, guidance on the latest security threats and defensive techniques, and training within the development organization. This group also is responsible for driving VMware products through external security accreditations and certifications.

VMware vSphere 5.0 and VMware vCenter Server 5.0 achieved Common Criteria certification at Evaluation Assurance Level 4 (EAL4+) under the CCEVS. Common Criteria is an international set of guidelines (ISO 15408) that provides a common framework for evaluating security features and capabilities of information technology (IT) security products. See Common Criteria Levels.

Prior to the reform in the Common Criteria certification scheme implemented in March 2012, EAL4+ was the highest assurance level recognized globally by all signatories under the Common Criteria Recognition Agreement (CCRA). See Common Criteria Reform.

Going forward, Common Criteria evaluations will transition from EAL levels to protection profiles. Over time, as protection profiles become available, EAL levels will disappear and products will be evaluated against the security requirements baseline defined by each protection profile. There is a virtualization profile in development to which future vSphere evaluations will conform. See Common Criteria Protection Profiles.

Until protection profiles become available for each specific technology type, security target–based evaluations will be performed at the EAL2 level. vSphere 5.1 is being evaluated at EAL2+. For more details on EAL4 versus EAL2 and protection profiles as they pertain to VMware vSphere 5.1 and other VMware products, see the VMware Common Criteria Update blog.

The VMware Security Response Policy documents the VMware commitment to resolving possible vulnerabilities in VMware products to assure customers that any such issues will be quickly corrected. See VMware Security Response Policy.

The VMware Security Center is a one-stop shop for security-related issues involving VMware products. It helps customers keep current on all security issues and to understand matters related to securing the virtual infrastructure. See VMware Security Center.

Conclusion

With VMware vSphere, VMware has built one of the most secure and robust virtualization platforms available. VMware has both the technology and the processes to ensure that this high standard is maintained in all current as well as future products. Users can be assured that all the benefits of running their most critical services on vSphere—specifically on VMware ESXi—do not come at the cost of security.

About the Author

Mike Foley is a senior technical marketing manager at VMware. His primary focus is on the security of VMware vSphere, the core platform. He is the author of the *vSphere 5.5 Security Hardening Guide*. His principal goal is to help IT and VI administrators build more-secure platforms that stand up to the scrutiny of security teams. Previously, Mike was on the evangelist team at RSA, where he concentrated on virtualization and cloud security and also served as a member of the product architect team. Mike contributes to the VMware vSphere and security blogs and has spoken on the topic of virtualization security at EMC World, RSA Conference, and VMworld®, as well as at other VMware events. Mike was awarded a patent (8,601,544) in December 2013 for dual-band authentication using the virtual infrastructure.

References

Understanding Full Virtualization, Paravirtualization, and Hardware Assist
http://www.vmware.com/resources/techresources/1008

Virtualization Overview
http://www.vmware.com/pdf/virtualization.pdf

vSphere Resource Management Guide, Storage I/O Control Resource Shares and Limits
http://pubs.vmware.com/vsphere-51/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-51-resource-management-guide.pdf

Understanding Memory Resource Management in VMware vSphere® 5.0
http://www.vmware.com/files/pdf/perf-vsphere-memory_management.pdf

Performance Best Practices for VMware vSphere® 5.1
http://www.vmware.com/resources/techresources/10329

Software and Hardware Techniques for x86 Virtualization
http://www.vmware.com/resources/techresources/10036

VMware Virtual Networking Concepts
http://www.vmware.com/files/pdf/virtual_networking_concepts.pdf

Intel® Virtualization Technology for Directed I/O
http://www.intel.com/content/dam/www/public/us/en/documents/product-specifications/vt-directed-io-spec.pdf

Intel® Virtualization Technology FlexMigration Enablement
https://communities.intel.com/servlet/JiveServlet/previewBody/5062-102-1-8119/Virtualization%20White%20Paper%200510.pdf

Software Techniques for Avoiding Hardware Virtualization Exits
http://labs.vmware.com/academic/publications/software-techniques

A Comparison of Software and Hardware Techniques for x86 Virtualization
https://www.vmware.com/pdf/asplos235_adams.pdf

VMware Storage
http://pubs.vmware.com/vsphere-55/topic/com.vmware.vsphere.storage.doc/GUID-F602EB17-8D24-400A-9B05-196CEA66464F.html

Common Criteria Reform
http://blogs.vmware.com/security/2013/04/vmware-common-criteria-update-april-2013.html

Common Criteria Levels
http://www.vmware.com/support/support-resources/certifications.html

Common Criteria Approved Protection Profiles
https://www.niap-ccevs.org/pp/

Common Criteria Draft Protection Profiles
https://www.niap-ccevs.org/pp/draft_pps/

VMware Common Criteria Update Blog
http://blogs.vmware.com/security/2013/04/vmware-common-criteria-update-april-2013.html

VMware Security Response Policy
http://www.vmware.com/vmtn/technology/security/security_response.html

VMware Security Center
http://www.vmware.com/security

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com
Copyright © 2014 VMware, Inc. All rights reserved. This product is protected by U.S. and international copyright and intellectual property laws. VMware products are covered by one or more patents listed at http://www.vmware.com/go/patents. VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions. All other marks and names mentioned herein may be trademarks of their respective companies. Item No: VMW-WP-SECRTY-vSPHR-HYPRVSR-USLET-101 Docsource: OIC - 14VM005.07