WHITE PAPER

Security Design of the VMware Infrastructure 3 Architecture

vmware®

Table of Contents

Introduction..3
VMware Infrastructure Architecture and Security Features 4
Virtualization Layer 4
CPU Virtualization...4
Memory Virtualization..5
Virtual Machines 6
Service Console..6
Virtual Networking Layer 7
Virtual Switches 7
Virtual Switch LANs 7
Virtual Ports 8
Virtual Network Adapters...8
Virtual Switch Isolation 8
Virtual Switch Correctness...9
Virtualized Storage...9
SAN Security..9
VMware VirtualCenter 10
Conclusion 11
About the Author...11
Acknowledgements 11
References..11

Security Design of the VMware Infrastructure 3 Architecture

Introduction

VMware Infrastructure is the most widely deployed software suite for optimizing and managing IT environments through virtualization — from the desktop to the data center. The only production-ready virtualization suite, VMware Infrastructure is proven at more than 20,000 customers of all sizes, used in a wide variety of environments and applications. VMware Infrastructure delivers transformative cost savings as well as increased operational efficiency, flexibility, and IT service levels.

VMware Infrastructure incorporates a number of features that directly address the security concerns of the most demanding datacenter environments. Some of these include:

- A virtualization layer designed from the ground up to run virtual machines in a secure manner while still providing high performance
- Compatibility with SAN security practices. VMware Infrastructure enforces security policies with LUN zoning and LUN masking.
- Implementation of secure networking features. VLAN tagging enhances network security by tagging and filtering network traffic on VLANs, and Layer 2 network security policies enforce security for virtual machines at the Ethernet layer in a way that is not available with physical servers.
- Integration with Microsoft® Active Directory. VMware Infrastructure allows you to base access controls on existing Microsoft Active Directory authentication mechanisms.

VMware Infrastructure 3, the latest generation of VMware datacenter products, includes several key enhancements that further address the security needs and challenges of modern IT organizations:

- Custom roles and permissions. VMware Infrastructure enhances security and flexibility with user-defined roles. You can restrict access to the entire inventory of virtual machines, resource pools and servers by assigning users to these custom roles.
- Resource pool access control and delegation. VMware Infrastructure secures resource allocation at different levels in the company. For example, when a top-level administrator makes a resource pool available to a department-level user, all virtual machine creation and management can be performed by the department administrator within the boundaries assigned to the resource pool.
- Audit trails. VMware Infrastructure maintains a record of significant configuration changes and the administrator who initiated each one. You can export reports for event tracking.
- Session management. VMware Infrastructure enables you to discover and, if necessary, terminate VirtualCenter user sessions.

The foundation for these features is a robust and secure virtualization architecture. VMware realizes that along with the benefits of virtualization comes the responsibility to ensure that a virtualized environment is as secure as can be, while still allowing the flexibility and benefits that advanced virtualization capabilities can provide.

VMware has implemented internal processes to ensure VMware products meet the highest standards for security. The VMware Security Response Policy (*www.vmware.com/vmtn/technology/security/security_response.html*) documents VMware's commitments for resolving possible vulnerabilities in VMware products so that customers can be assured that any such issues will be corrected quickly. The VMTN Security Center (*www.vmware.com/security*) is a one-stop shop for security-related issues involving VMware products. It helps you stay up-to-date on all current security issues and to understand considerations related to securing your virtual infrastructure.

The success of this architecture in providing a secure virtualization 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, VMware ESX Server 2.5.0 and VirtualCenter 1.2.0 have been validated under the U.S. Common Criteria Evaluation and Validation Scheme (CCEVS) process, achieving EAL2 certification. VMware ESX Server 3.0 and VirtualCenter 2.0 are currently being tested for certification at EAL4+.

The rest of this document discusses the architecture of ESX Server 3 and VirtualCenter 2, focusing on the security aspects of the design.

VMware Infrastructure Architecture and Security Features

From a security perspective, VMware Infrastructure consists of several major components:

- The virtualization layer, consisting of the VMkernel and the virtual machine monitor
- The virtual machines
- The ESX Server service console
- The ESX Server virtual networking layer
- Virtual storage
- VirtualCenter

Virtualization Layer

VMware ESX Server presents a generic x86 platform by virtualizing four key hardware components: processor, memory, disk, and network. An operating system is then installed into this virtualized platform. The virtualization layer, or VMkernel, is a kernel designed by VMware specifically to run virtual machines. It controls the hardware utilized by ESX Server hosts and schedules the allocation of hardware resources among the virtual machines. Because the VMkernel is fully dedicated to supporting virtual machines and is not used for other purposes, the interface to the VMkernel is strictly limited to the API required to manage virtual machines. There are no public interfaces to VMkernel, and it cannot execute arbitrary code.

The VMkernel alternates among all the virtual machines on the host in running the virtual machine instructions on the processor. Every time a virtual machine's execution is stopped, a context switch occurs. During the context switch the processor register values are saved and the new context is loaded. When a given virtual machine's turn comes around again, the corresponding register state is restored.

Each virtual machine has an associated virtual machine monitor. (VMM) The VMM uses binary translation to modify the guest operating system kernel code so it can run in a less-privileged processor ring. This is analogous to what a Java virtual machine does using just-in-time translation. Additionally, the VMM virtualizes a chip set for the guest operating system to run on. The device drivers in the guest cooperate with the VMM to access the devices in the virtual chip set. The VMM passes request to the VMkernel to complete the device virtualization and support the requested operation.

Note: The VMM utilized by ESX Server is the same as the one used by other VMware products that run on host operating systems, such as VMware Workstation. Therefore, all comments related to the VMM apply to all VMware virtualization products.

CPU Virtualization

Binary translation is a powerful technique that can provide CPU virtualization with high performance. The VMM uses a translator with the following properties:

- **Binary** — Input is binary x86 code, not source code.
- **Dynamic** — Translation happens at run time, interleaved with execution of the generated code.
- **On demand** — Code is translated only when it is about to execute. This eliminates need to differentiate code and data.
- **System level** — The translator makes no assumptions about the code running in the virtual machine. Rules are set by the x86 architecture, not by a higher-level application binary interface.
- **Subsetting** — The translator's input is the full x86 instruction set, including all privileged instructions; output is a safe subset (mostly user-mode instructions).
- **Adaptive** — Translated code is adjusted in response to virtual machine behavior changes to improve overall efficiency.

During normal operation, the translator reads the virtual machine's memory at the address indicated by the virtual machine program counter, classifying the bytes as prefixes, opcodes, or operands to produce intermediate representation objects. Each intermediate representation object represents one guest instruction. The translator accumulates intermediate representation objects into a translation unit, stopping at 12 instructions or a terminating instruction (usually flow control).

Buffer overflow attacks usually exploit code that operates on unconstrained input without doing a length check. The classical example is a string that represents the name of something — a file, for example. If it is possible to provide a very, very long string and the code that operates on the string (that is, the filename) has a fixed size buffer, and it does not perform length checks, a buffer overflow occurs and may be used in an attack. Since the binary translator does not operate on translation units of more than 12 instructions, it is not possible for the translator to experience a buffer overflow for this operation.

Similar design principles are applied throughout the VMM code. There are few places where the VMM operates on data specified by the guest operating system, so the scope for buffer overflows is much smaller than in a general-purpose operating system. In addition, VMware programmers develop the software with awareness of the importance of programming in a secure manner. This approach to software development greatly reduces the chance that vulnerabilities will be overlooked. To provide an extra layer of security, the VMM supports the buffer overflow prevention capabilities built in to most Intel and AMD CPUs, known as the NX or XD bit.

Intel's hyperthreading technology allows two process threads to execute on the same CPU package. These threads can share the memory cache on the processor. Malicious software can exploit this feature by having one thread monitor the execution of another thread, possibly allowing theft of cryptographic keys. ESX Server virtual machines do not provide hyperthreading technology to the guest operating system. ESX Server, however, can utilize hyperthreading to run two different virtual machines simultaneously on the same physical processor. However, because virtual machines do not necessarily run on the same processor continuously, it is more challenging to exploit the vulnerability discussed above. However, if you want a virtual machine to be protected against the small chance of the type of attack discussed above, ESX Server provides an option to isolate a virtual machine from hyperthreading. VMware knowledge base article 1728 provides further details on this topic.

Hardware manufacturers have begun to incorporate CPU virtualization capabilities into processors. Although the first generation of these processors does not perform as well as VMware's software-based binary translator, VMware will continue to work with the manufacturers and make appropriate use of their technology as it evolves.

Memory Virtualization

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

The operating system that executes within a virtual machine expects a zero-based physical address space, as provided by real hardware. 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, while 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 highlight this deviation from the usual meaning of the term.

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 operating system page tables or translation lookaside 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 permits ordinary memory references to execute without additional overhead, since the hardware translation lookaside buffer caches direct virtual-to-machine address translations read from the shadow page table. As memory management capabilities are enabled in hardware, VMware will take full advantage of the new capabilities while maintaining the same strict adherence to isolation.

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 allows the VMM to interpose on guest memory accesses. Any attempt by the operating system or any application running inside a virtual machine to address memory outside of what has been allocated by the VMM would cause a fault to be delivered to the guest operating system, typically resulting in an immediate system crash, panic, or halt in the virtual machine, depending on the operating system. This is often termed hyperspacing, when a malicious guest operating system attempts I/O to an address space that is outside normal boundaries.

When a virtual machine needs memory, each memory page is zeroed out by the VMkernel 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 see it. The exception is when transparent page sharing is in effect.

Transparent page sharing is a technique for using memory resources more efficiently. 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 operating system. As soon as any one virtual machine tries 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.

Virtual Machines

Virtual machines are the containers in which guest operating systems and their applications run. By design, all VMware virtual machines are isolated from one another. Virtual machine isolation is imperceptible to the guest operating system. Even a user with system administrator privileges or kernel system level access on a virtual machine's guest operating system cannot breach this layer of isolation to access another virtual machine without privileges explicitly granted by the ESX Server system administrator.

This isolation enables multiple virtual machines to run securely while sharing hardware and ensures both their ability to access hardware and their uninterrupted performance. For example, if a guest operating system running in a virtual machine crashes, other virtual machines on the same ESX Server host continue to run. The guest operating system crash has no effect on:

- The ability of users to access the other virtual machines
- The ability of the running virtual machines to access the resources they need
- The performance of the other virtual machines

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

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. 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 ESX Server host only through a virtual switch. Further, a virtual machine communicates with the physical network, including virtual machines on other ESX Server hosts, only through a physical network adapter.

In considering virtual machine isolation in a network context, you can apply these rules:

- 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.
- If no physical network adapter is configured for a virtual machine, the virtual machine is completely isolated from any physical networks.
- If you use the same safeguards (firewalls, antivirus software, and so forth) to protect a virtual machine from the network as you would for a physical machine, the virtual machine is as secure as the physical machine would be.

You can further protect virtual machines by setting up resource reservations and limits on the ESX Server host. For example, through the fine-grained resource controls available in ESX Server, you can configure a virtual machine so that it always gets at least 10 percent of the ESX Server host's CPU resources, but never more than 20 percent. Resource reservations and limits protect virtual machines from performance degradation if another virtual machine tries to consume too many resources on shared hardware. For example, if one of the virtual machines on an ESX Server host is incapacitated by a denial-of-service or distributed denial-of-service attack, a resource limit on that machine prevents the attack from taking up so many hardware resources that the other virtual machines are also affected. Similarly, a resource reservation on each of the virtual machines ensures that, in the event of high resource demands by the virtual machine targeted by the denial-of-service attack, all the other virtual machines still have enough resources to operate.

By default, ESX Server imposes a form of resource reservation by applying a distribution algorithm that divides the available host resources equally among the virtual machines while keeping a certain percentage of resources for use by system components, such as the service console. This default behavior provides a degree of natural protection from denial-of-service and distributed denial-of-service attacks. You set specific resource reservations and limits on an individual basis if you want to customize the default behavior so the distribution is not equal across all virtual machines on the host.

Service Console

The ESX Server 3.0 service console provides an execution environment to monitor and administer the entire ESX Server host. The service console operating system is a reduced version of Red Hat Enterprise Linux 3, Update 6. Because it has been stripped of functionality not necessary for interacting with the ESX Server 3 virtualization layer, not all vulnerabilities of this distribution apply to the service console. VMware monitors and tracks all known security exploits that apply to this particular reduced version and issues custom updates as and when needed.

If the service console is compromised, the virtual machines it interacts with might also be compromised. This is analogous to an intruder gaining access to the ILOM service console of a physical server. To minimize the risk of an attack through the service console, ESX Server protects the service console with a firewall. In addition, here are some of the other ways ESX Server minimizes risks to the service console:

- ESX Server runs only services essential to managing its functions, and the Linux distribution is limited to the features required to run ESX Server.

- By default, ESX Server is installed with a high security setting, which means that all outbound ports are closed and the only inbound ports that are open are those required for interactions with clients such as the VMware Virtual Infrastructure Client. VMware recommends that you keep this security setting unless the service console is connected to a trusted network.
- All communications from clients are encrypted through SSL by default. The SSL connection uses 256-bit AES block encryption and 1024-bit RSA key encryption.
- The Tomcat Web service, used internally by ESX Server to support access to the service console by Web clients such as VMware Virtual Infrastructure Web Access, has been modified to run only those functions required for administration and monitoring by a Web client.
- VMware monitors all security alerts that could affect service console security and, if needed, issues a security patch, as it would for any other security vulnerability that could affect ESX Server hosts. VMware provides security patches for Red Hat Enterprise Linux 3, Update 6 and later as they become available
- Insecure services such as FTP and Telnet are not installed and the ports for these services are closed by default.
- The number of applications that use a setuid or setgid flag has been minimized.
- ESX Server supports SNMPv1, and the management information base is read-only. Nothing can be set through SNMP management calls.

Although the service console provides an avenue by which virtual machines can be manipulated, ESX Server is designed to enable the administrator to place it on an entirely isolated internal network, via a separate VLAN or even using an entirely separate network adapter. Thus the risk of compromise is one that can be managed in a straightforward way.

Virtual Networking Layer

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

Virtual Switches

The networking stack was completely rewritten for ESX Server 3 using a modular design for maximum flexibility. A virtual switch is "built to order" at run time from a collection of small functional units, such as:

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

When the virtual switch is built at run time, ESX Server loads only those components it needs. It installs and runs only what is actually needed to support the specific physical and virtual Ethernet adapter types used in the configuration. This means the system pays the lowest possible cost in complexity and hence makes the assurance of a secure architecture all the more possible.

Virtual Switch VLANs

ESX Server supports IEEE 802.1q VLANs, which you can use to further protect the virtual machine network, service console, or storage configuration. This driver is written by VMware software engineers according to the IEEE specification. VLANs let you segment a physical network so that 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) — You may install an 802.1Q VLAN trunking driver inside the virtual machine, and 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) — You may use external switches for VLAN tagging. This is similar to a physical network, and VLAN configuration is normally transparent to each individual physical server.
- Virtual switch tagging (VST mode) — In this mode, you provision one port group on a virtual switch for each VLAN, then attach the virtual machine's virtual adapter to the port group instead of the virtual switch directly. 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 a different VLAN.

Virtual Ports

The virtual ports in ESX Server provide a rich control channel for communication with the virtual Ethernet adapters attached to them. ESX Server virtual ports know authoritatively what are the configured receive filters for virtual Ethernet adapters attached to them. This means no learning is required to populate forwarding tables.

Virtual ports also know authoritatively 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 know for sure what is "burned into ROM" (actually, stored in the configuration file, outside control of the guest operating system).

The policies available in virtual ports are much harder to implement — if they are possible at all — with physical switches. Either someone must manually program the ACLs into the switch port, or you must rely on such weak assumptions as "first MAC seen is assumed to be correct."

The port groups used in ESX Server do not have a counterpart in physical networks. You can think of them as templates for creating virtual ports with particular sets of specifications. Because virtual machines move around from host to host, ESX Server needs 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 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

Thus, port groups provide a powerful way to define and enforce security policies for virtual networking.

Virtual Network Adapters

VMware Infrastructure 3 provides several types of virtual network adapters that guest operating systems can use. The choice of adapter depends upon factors such as support by the guest operating system and performance, but all of them share these 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 you power on the virtual machine on which the adapters are configured, when you take an explicit action to connect the device, or when you migrate a virtual machine using 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 may ignore any requests from the virtual Ethernet adapter that would violate the level 2 security policy in effect for the port.

Virtual Switch Isolation

A common cause of traffic leaks in the world of physical switches is cascading — often needed because physical switches have a limited number of ports. Because virtual switches provide all the ports you need in one switch, there is no code to connect virtual switches. ESX Server provides no path for network data to go between virtual switches at all. Therefore, it is relatively easy for ESX Server to avoid accidental violations of network isolation or violations that result from malicious software running in a virtual machine or a malicious user. In other words, the ESX Server system does not have complicated and potentially failure-prone logic to make sure that only the right traffic travels from one virtual switch to another; instead, it simply does not implement any path that any traffic could use to travel between virtual switches. Furthermore, virtual switches cannot share physical Ethernet adapters, so there is no way to fool 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 allow 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.

A would-be attacker would likely have to find a remote code execution bug in the vmkernel to circumvent virtual switch isolation. Because ESX Server parses so little of the frame data — primarily just the Ethernet header — this would be difficult.

There are natural limits to this isolation. If you connect the uplinks of two virtual switches together, or if you bridge two virtual switches with software running in a virtual machine, you open the door to the same kinds of problems you might see in physical switches.

Virtual Switch Correctness

It is important to ensure that virtual machines or other nodes on the network cannot affect the behavior of the virtual switch. ESX Server guards against such influences in the following ways:

- Virtual switches do not learn from the network in order to populate their forwarding tables. This eliminates a likely vector for deinal-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 important to ensure that frames are contained within the appropriate VLAN on a virtual switch. ESX Server 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 really just a special case of the general principle that the system should not trust user accessible data.
- Virtual switches have no dynamic trunking support.
- Virtual switches have no support for what is referred to as native VLAN.

Dynamic trunking and native VLAN are features in which an attacker may find vulnerabilities that could open isolation leaks. This is not to say that these features are inherently insecure, but even if they are implemented securely, their complexity may lead to misconfiguration and open an attack vector.

Virtualized Storage

ESX Server 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 operating system first goes to the appropriate driver in the virtual machine. For storage controllers, ESX Server emulates LSI Logic or BusLogic SCSI devices, so the corresponding driver loaded into the guest operating system is either an LSI Logic or a BusLogic driver. The driver 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 virtual machine monitor, then handled by device emulation code in the virtual machine monitor based on the specific I/O port being accessed. The virtual machine monitor then calls device-independent network or disk code to process the I/O. For disk I/O, ESX Server 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 ESX Server for the specific device on the physical machine.

SAN Security

A host running ESX Server is attached to a Fibre Channel SAN in the same way that any other host is. 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 operating system, but for ESX Server, the drivers are installed in the ESX Server VMkernel. ESX Server also includes VMware Virtual Machine File System (VMware VMFS), a distributed file system and volume manager that creates and manages virtual volumes on top of the LUNs that are presented to the ESX Server host. Those virtual volumes, usually referred to as virtual disks, are allocated to specific virtual machines.

Virtual machines have no knowledge or understanding of Fibre Channel. The only storage available to virtual machines is on SCSI devices. Put another way, a virtual machine does not have virtual Fibre Channel HBAs but only has virtual SCSI adapters. Each virtual machine is able to see only the virtual disks that are presented to it on its virtual SCSI adapters. This isolation is complete, with regard to both security and performance. A VMware virtual machine has no visibility into the WWN (world wide name), the physical Fibre Channel HBAs, or even the target ID or other information about the LUNs upon which its virtual disks reside. The virtual machine is isolated to such a degree that software executing in the virtual machine cannot even detect that it is running on a SAN fabric. Even multipathing is handled in a way that is transparent to a virtual machine. Furthermore, virtual machines can be configured to limit 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 taking over the Fibre Channel HBA.

Consider the example of running a Microsoft Windows operating system inside a VMware virtual machine. The virtual machine sees only the virtual disks chosen by the ESX Server administrator at the time the virtual machine is configured. This operation of configuring a virtual machine to see only certain virtual disks is effectively LUN masking in the virtualized environment. It has the same security benefits as LUN masking in the physical world; it is just done with a different set of tools. Software executing in the virtual machine — including the Windows operating system — is aware of only the virtual disks attached to the virtual machine. Even if the Windows operating system attempts to issue a SCSI command — Report LUNs, for example — in an effort to discover other targets, ESX Server 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 ESX Server hosts is accessing common targets or LUNs. The VMware VMFS file system ensures that all of the hosts in the cluster cooperate to ensure correct permissions and safe access to the VMware VMFS volumes. File locks are stored on disk as part of the volume metadata, and all ESX Server hosts utilizing the volumes are aware of the ownership. Ownership of files and various distributed file system activities are rendered exclusive and atomic by the use of standard SCSI reservation primitives.

Each virtual disk (sometimes 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 ESX Server host is allowed to access that virtual disk. This situation does not change fundamentally when there is a cluster of ESX Server hosts, with multiple virtual machines powered on and accessing virtual disks on a single VMware VMFS volume. Because of this fact, VMotion — which enables live migration of a virtual machine from one ESX Server host to another — is a protected operation.

VMware VirtualCenter

VMware VirtualCenter provides a central place where almost all management functions of VMware Infrastructure can be performed. VirtualCenter relies on Windows security controls , and hence must reside on a properly managed server with network access limited to those ports necessary for it to interoperate with all the other VMware components. It is role-based and tied to Active Directory or legacy NT domains, which makes it unnecessary to create custom user accounts for it.. VirtualCenter also keeps records of nearly every event in the ESX Server system, allowing the generation of audit trails for compliance.

VirtualCenter manages the creation and enforcement of resource pools, which are used to partition available CPU and memory resources. A resource pool can contain child resource pools as well as virtual machines, allowing the creation of a hierarchy of shared resources. Resource pools allow you to delegate control over resources of a host or cluster. When a top-level administrator makes a resource pool available to a department-level administrator, that administrator can then perform all virtual machine creation and management within the boundaries of the resources to which the resource pool is entitled. More important, VirtualCenter enforces isolation between resources pools, so that resource usage in one pool does not affect availability of resources in another pool. This provides a coarser level of granularity for containment of resource abuse, in addition to that provided on the ESX Server host level.

VirtualCenter has a sophisticated system of roles and permissions, to allow fine-grained determination of authorization for administrative and user tasks, based on user or group and inventory item, such as clusters, resource pools, and hosts. This system allows you to insure that only the minimum necessary privileges are assigned to people in order to prevent unauthorized access or modification.

VirtualCenter Server uses X.509 certificates to encrypt session information sent over SSL (secure sockets layer protocol) connections between server and client components. You can replace all default self-signed certificates generated at installation time with legitimate certificates signed by your local root certificate authority or public, third-party certificates available from multiple public certificate authorities.

VirtualCenter asks for root credentials when it first connects to an ESX Server host. The root password for that host is cached only long enough to enable VirtualCenter management functionality, and the communication channel to the host is encrypted. VirtualCenter then creates a user called vpxuser with a pseudo-randomly generated password and uses the vpxuser account for subsequent connections and management operations. The vpxuser account for each ESX Server host has a unique, 32-character (256-bit) password that is generated from a cryptographically random string of data that is mapped to a set of legal password characters. Once generated, the password is encrypted using 1024-bit RSA key encryption. For encryption details, you can examine the certificate in the Documents and Settings folder that is applicable to VirtualCenter (usually at `C:\Documents and Settings\AllUsers\ApplicationData\VMware\VMwareVirtualCenter\SSL`). The password is also stored encrypted on the host, as any local account password would be (see `man 3 crypt` in the service console on an ESX Server host for details).

The vpxuser account is created for VirtualCenter management when a host is added to VirtualCenter and is used only to authenticate the connection between VirtualCenter and the ESX Server host. Entries corresponding to the account are added to `/etc/passwd` and `/etc/shadow`, but no process actually runs as vpxuser on ESX Server.

The vpxuser password is reset every time a host is added to VirtualCenter. If VirtualCenter is disconnected from a host, it tries to reconnect with the vpxuser and password that is stored encrypted in the VirtualCenter database. If that fails, the user is prompted to reenter the root password so the system can reset (that is, automatically generate a new password for the vpxuser account).

In the VirtualCenter code, database specific variable protection mechanisms, such as parameterized queries in SQL Server are used extensively, thereby greatly reducing the risk of any SQL injection attack. The VIM API, which is the main SDK library, allows for a mechanism to specify privileges necessary to invoke the API as part of the API definition. This ensures that security implications are taken into consideration from the beginning of writing a new API.

Conclusion

With VMware Infrastructure 3, 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 and future products. You can be assured that all the benefits of running your most critical services on VMware Infrastructure 3 do not come at the cost of the security of your environment.

About the Author

Charu Chaubal is technical marketing manager at VMware, where he specializes in enterprise datacenter management. Previously, he worked at Sun Microsystems, where he had more than seven years' experience designing and developing distributed resource management and grid infrastructure software solutions. He has also developed and delivered training courses on grid computing to a variety of customers and partners in the United States and abroad. Chaubal received a Bachelor of Science in Engineering from the University of Pennsylvania and a Ph.D. from the University of California at Santa Barbara, where he studied the numerical modeling of complex fluids. He is the author of numerous publications and several patents in the fields of datacenter automation and numerical price optimization.

Acknowledgements

The author would like to thank the following for their valuable input: Ole Agesen, Ben Cheung, Mukund Gunti, Kirk Larsen, Ophir Rachman, and Andrew Sharpe.

References

K. Adams and O. Agesen, A Comparison of Software and Hardware Techniques for x86 Virtualization. ASPLOS October 2006.
www.vmware.com/vmtn/resources/528

Common Criteria Evaluation and Validation Scheme Validation Report for VMware ESX Server 2.5.0 and VirtualCenter 1.2.0, Report Number: CCEVS-VR-06-0013, March 27, 2006
niap.bahialab.com/cc-scheme/st/ST_VID10056.cfm

ESX Server Architecture and Performance Implications
www.vmware.com/vmtn/resources/433

Foundstone Professional Services, VMware ESX Server Software Security Assessment, December 2006

Foundstone Professional Services, VMware VirtualCenter Software Security Assessment, December 2006

Michael Nelson, Beng-Hong Lim, and Greg Hutchins, Fast Transparent Migration for Virtual Machines, *Proceedings of USENIX '05: General Track*, Anaheim, California, USA, April 2005
www.vmware.com/pdf/usenix_vmotion.pdf

Providing LUN Security
www.vmware.com/vmtn/resources/425

VMware ESX Server: Third-Party Software in the Service Console
www.vmware.com/vmtn/resources/516

VMware Infrastructure 3 Architecture
www.vmware.com/vmtn/resources/410

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

Carl A. Waldspurger, Memory Resource Management in VMware ESX Server, *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation* (OSDI '02), Boston, Massachusetts, December 2002
www.vmware.com/pdf/usenix_resource_mgmt.pdf

Revision: 20070215 Item: WP-013-PRD-01-01

vmware®