Security Assurance Requirements for Hypervisor Deployment Features

Ramaswamy Chandramouli
Computer Security Division, Information Technology Laboratory
National Institute of Standards & Technology
Gaithersburg, MD, USA
mouli@nist.gov

***Abstract*-Virtualized hosts provide abstraction of the hardware resources (i.e., CPU, Memory, etc.) enabling multiple computing stacks to be run on a single physical machine. The Hypervisor is the core software that enables this virtualization and hence must be configured to ensure security robustness for the entire virtualization infrastructure. Among the various combination of hypervisor types and hypervisor hardware platforms, we have chosen a reference architecture as the basis for our set of deployment features. For each deployment feature, this paper looks at the configuration options and analyzes the security implications of the options/deployment feature to derive a set of assurance requirements that are either (a) provided by each of the configuration options; or (b) required for that deployment feature as a whole regardless of configuration options.**

Keywords-Virtual Machine; Virtual Network; Hypervisor; Virtualized Host; Security Assurance Requirements

I. INTRODUCTION

Virtualized hosts provide abstraction of the underlying hardware resources to enable multiple computing stacks (consisting of an O/S, Middleware and Applications) to be run on a single physical host. Because of many beneficial features such as efficient utilization of hardware resources, elasticity, flexibility and in some instances better security, virtualized hosts are being increasingly deployed in many data centers built for in-house enterprise use or offering cloud-based services.

The core software that provides the virtualization capabilities in a virtualized host is called the Hypervisor. The hypervisor provides the following major functions: (a) Abstraction of all underlying hardware resources (e.g., CPU, Memory, etc.). This enables multiple computing stacks called Virtual Machines (VMs) (each with its own different brand of O/S) to be run on a single physical machine; (b) Isolation of run-time process stack in one VM from another; (c) Selective connectivity or communication among VMs through a suitable network configuration inside the virtualized host (called Virtual Network); and (d) Sharing of hardware resources statically through pre-defined resource limits and dynamically through multiplexing/scheduling features [1]. These functions together with some associated housekeeping functions could broadly be classified under two feature sets: (a) Hardware Abstraction feature set and (b) Virtual Machine Management feature set. The virtual machine management feature encompasses all functions relating to the life cycle of VMs – create, stop, suspend, activate etc.

The hypervisor software as a software entity as well as in combination with its hardware platform can have different architectures. For example, both of the two major feature sets could be provided by a monolithic software module or they could be split between two modules – with the first module called Hypervisor providing just the hardware abstraction function while a separate module called Virtual Machine Manager (VMM) provides the virtual machine management function [2]. A further variation in the case of those hypervisor architectures with a separate VMM module is that the VMM module can be run as a separate protected VM with higher privileges than other VMs (usually called Guest VMs) [3]. Some of them are directly installed on the hardware (or bare metal) (Type 1), while some need an operating system (called a host operating system) on the physical host to be installed (Type 2) [2]. There could also be variation in terms of whether the platform provides hardware assistance for virtualization or not. Hardware-assisted features for virtualization include the availability of two execution modes (i.e., root mode and non-root mode) and multiple privilege rings (i.e., enabling different commands to run at different privilege levels) in addition to memory management features (e.g., nested page table or extended page table, etc.). The consequence of the hardware providing some virtualization functions is that the corresponding hypervisor module can be thin (enabling better security verification/attestation) and, at the same time, be able to provide a feature called full virtualization (enabling guest VMs to run unmodified versions of

 ISBN: 978-1-61208-249-3

commercial O/S offerings instead of a version that is specially modified and ported to run on virtualized platforms) [3].

The reference hypervisor platform we have chosen for security analysis in this paper consists of a Type 1 hypervisor that provides full virtualization with either a monolithic or two-piece software module. For each deployment feature pertaining to this architecture, we look at the configuration options available for enabling that feature. We then analyze the security implications of these options/deployment features to derive a set of security assurance requirements that are either (a) provided by each of the configuration options, or (b) required for that deployment feature as a whole regardless of configuration options.

II. HYPERVISOR LOCAL USER MANAGEMENT AND AUTHENTICATION

All commercial hypervisor offerings come with a Management server, almost eliminating the need for creating local users and groups on each hypervisor (virtualized) host. However, some tasks cannot be accomplished through the management server [4] alone, such as the need to troubleshoot the hypervisor boot and configuration problems and the need to audit the hypervisor host configuration and remote access. In spite of the need for local users and groups, it is a good practice to restrict the number of users to just two or three. A local user on a hypervisor always performs only administrative functions and is not a typical business application end-user.

With respect to local user management and authentication on a hypervisor, the two options are:

(a) Manage the users and groups associated with a hypervisor locally;

(b) Manage the users and groups by integrating with a local directory infrastructure (e.g., Active Directory) and using a directory-based authentication mechanism (e.g., Kerberos).

The security analysis of these options is given below:

A. *Managing the users and groups associated with a hypervisor locally:*

Local users and groups for the hypervisor host are usually created using a service console (if the hypervisor architecture includes one) or through a dedicated client interface. The programs for this usually include features to set a password for the user account as well as options to set basic access mode permissions (e.g., SSH, VPN, etc.) [4].

Security issues associated with managing users and groups locally through the manual process are:

(a) When an administrative user having user accounts on some hypervisor hosts, quits the organization or moves over to a different division within the company, the user account associated with him/her has to be manually deleted in all hypervisor hosts. If not done properly, it can leave zombie accounts which can be exploited resulting in a security breach of the hypervisor host [5].

(b) Any changes to organization policy such as the password policy has to be enforced manually on each hypervisor leaving room for some mismatches in some hypervisor hosts.

B. Managing the users and groups through a Directory Infrastructure

In this option, the users and groups are still created through either the service console command line interface or through a dedicated hypervisor client but there are three differences [4]:

(a) The administrative user account names created here match those already present in the enterprise directory.

(b) No passwords are assigned while creating these user accounts.

(c) A suitable command modifies the configuration to specify that the authentication will take place through a mechanism appropriate for the directory infrastructure (e.g., through the domain controller in an Active Directory infrastructure using an authentication mechanism such as Kerberos).

The advantages in managing the users of the hypervisor host through the directory infrastructure are:

(a) User account changes (such as deletion) can be done centrally at the directory level. This way, an account for a user no longer with the organization, though still present in the hypervisor host, cannot be used for logging in, since password and forms of authentication have to be done at the directory infrastructure level. The latter will fail since the user account no longer exists there [4];

(b) Password policies such as complexity, expiration times, etc., can be centrally defined and enforced; and

(c) Robust Authentication mechanisms can be set up because of integration of authentication function with the directory infrastructure that is not available locally in the hypervisor host.

III. HYPERVISOR CPU SCHEDULER CONFIGURATION

Most hypervisors provide the following configuration options for sharing the physical CPU of the virtualized host

 ISBN: 978-1-61208-249-3

among the multiple virtual CPUs of the VMs: (a) Guaranteed CPU time slots for VMs based on their assigned weights and (b) Fair-share scheduling where a VM gets physical CPU time based upon its weight but subject to a cap for the amount of CPU.

The proportional fair-share scheduling option is recommended for most VM workloads from both a load balancing and security point of view [6]. In scheduling options with time guarantees, an errant process in a rogue or compromised VM could hog all the CPU resources of the virtualized host resulting in **denial of service** to other VMs. However, for VMs running applications with critical response times, (e.g., process control application), the only scheduling option is the one that provides time guarantees.

IV. HYPERVISOR ACCESS CONTROL CONFIGURATION

There are two main classes of administrative operations (no user operations) in a hypervisor (virtualized) host: (a) Virtual Machine Management operations; and (b) Configuration of Virtual Network involving the VMs. These classes determine the overall deployment of the entire virtualized infrastructure. The granularity at which access control permissions can be set for these operations contributes towards the security robustness of the infrastructure. Our reference architecture for hypervisor platform admits instances where these operations are either performed by core hypervisor software (monolithic hypervisor architecture) or through an interface provided by a separate VMM module installed as a secure dedicated VM (e.g., Dom0 in Xen hypervisor and Parent Partition in Hyper-V hypervisor)[3].In both variations, the following architectural options for performing access control functions exist:

(a) Built-in Access Control Module: In this approach, there is a built-in access control module that is an integral part of the hypervisor executable.

(b) Pluggable Access Control Module: In this approach, a pluggable, external, custom access control module can be specified as a component of hypervisor kernel modules and then it can be booted together during the hypervisor boot-up. Using the interfaces provided by the module, policies are then defined based on the set of access control models (e.g., RBAC, MAC, Type Enforcement, etc.) supported by that module. This approach has been adopted in hypervisors such as Xen. Loading of custom access control modules requires the implementation of a generic security framework; the name of such a framework in the Xen context is known as Xen Security Module (XSM) [3].

Irrespective of the type of access control module, the module should provide the following features in two feature classes to obtain robust security for the virtual infrastructure.

(a) Feature Class – Aggregation: The access control module should provide capabilities for defining artifacts containing (i) arbitrary combination of users (Custom Group); (ii) arbitrary combination of permissions (Custom Role); (iii) combination of objects based on any administrator-defined logic (Custom Objects - e.g., set of VMs that house a Webserver, set of VMs that together form all tiers of a multitiered application); and (iv) A parent-child relationships using objects (Object Hierarchy).

(b) Feature Class – Permission Assignment Granularity: The access control module should provide the flexibility to assign permissions at various levels of granularities. The minimal requirements are: (i) All administrative permissions on a single object or a custom object (e.g., all life-cycle operations on a designated VM); (ii) A particular type of permission (e.g., view) on all objects (e.g., view the list of all VMs in the virtualized host but not exercise any other operations on those VMs); (iii) Arbitrary combination of permissions (contained in a Custom Role) on arbitrary combination of objects (Custom Object); and (iv) Have a set of permissions on an object at the top or middle of an object hierarchy but negation of those permissions for a specific child object (e.g., ability to take snapshots, start and stop all VMs except a designated VM running a sensitive application).

V. HYPERVISOR DEVICE DRIVERS CONFIGURATION

Device drivers are software pieces that provide access to a physical device such as a hardware drive or network interface card to guest VMs. Generally, these device drivers are either supplied by physical device vendors or written by third parties and hence are traditionally held as untrusted code. They form one of the weakest links in the security configuration of a virtualized host (hypervisor platform).

The following device driver configuration options depend upon the architecture of hypervisor platform [7]: (a) In the monolithic hypervisor architecture, the device driver module is an integral part of the code for the hypervisor build; and (b) In a hypervisor architecture with a separate VMM, the device drivers may be located either in the parent partition that houses the VMM module or it may

 ISBN: 978-1-61208-249-3

be run in a dedicated guest VM (e.g., Driver Domain in Xen hypervisor platform)[3].

The security assurance requirement for each of the above two device driver configuration options are:

(a) Some hypervisor offerings provide a feature to pre-define a secure configuration for the entire hypervisor platform installation by what are known as "Host Image Profiles" (e.g., VMware VSphere 4)[4]. Such a host image profile can be used to specify acceptable drivers (based on assessments by third-party certifiers) or specify an acceptance level for the device driver that is part of the hypervisor build.

(b) When device drivers are run in a VM different from the parent partition (VM that houses the VMM module), that VM should contain just the barebones guest O/S, the device driver software designated to run on it and any other software required for that VM to share the device with other domains. This configuration requirement is needed since the VM that runs the driver code that enables other VMs to share devices is theoretically part of the Trusted Computing Base of the hypervisor platform.

VI. HYPERVISOR MANAGEMENT INTERFACE CONFIGURATION

The hypervisor should be accessible and configured only through a dedicated management network. The management interface (also called VMKernel interface) of the hypervisor is accessed through a special port (or port group) called the VMKernel portgroup in some offerings [4]. By assigning this port or portgroup (with its associated VLAN ID) to its own virtual switch and connecting this virtual switch to a dedicated physical network interface card (pNIC) of the virtualized host (with a redundant pNIC standby), a dedicated management network can be created. There should also be restrictions on services (e.g., DNS) and network locations (e.g., IP addresses) that can interact with the management interface [8]. These restrictions can be defined and enforced through a firewall whose configuration has the following options:

(a) A firewall external to the virtualized host; or

(b) A firewall built into the hypervisor module

Irrespective of the firewall used, the following security checklist should apply:

(a)All incoming and outgoing traffic must be blocked, except those that are needed for the hypervisor management access. These include but are not limited to: SSH (TCP port 22), DNS (UDP port 53) and DHCP (UDP port 68).

(b) The previous configuration setting merely specifies the set of allowed services. It is also necessary to restrict the clients that can avail of these services. It is a well-understood practice that hypervisor management should be limited to a restricted set of (preferably local) IP addresses or range of IP addresses (subnet).

VII. HYPERVISOR VIRTUAL NETWORK CONFIGURATION

A virtual network is a network defined entirely within a single physical (virtualized) host; a typical configuration is given in Figure 1. It consists of software-defined virtual network interface cards (vNIC) associated with VM that are connected to software-defined virtual switches (vSwitch), which in turn are connected to the physical network interface cards (pNIC) of the virtualized host [9]. The vNICs and vSwitches are defined using the hypervisor management interface and they together with the network traffic flowing between them reside entirely in the memory of the virtualized host. Multiple portgroups (each with its associated virtual LAN ID) can be defined on a single vSwitch and each is connected to one or more vNICs on the VMs. Network traffic between VMs connected to the same vSwtich and portgroup never leaves the virtualized host. The virtual network thus enables communication among the VMs within the virtualized host as well as communication with the enterprise network outside the virtualized host.

The presence of this virtual network poses a threat to the hypervisor in the following ways [10]:

(a)A compromised application within a VM can attack the hypervisor

(b)An application within a VM can be used as a launching pad to compromise applications in other VMs on the same virtualized host.

To protect the hypervisor, use one of the following configuration options:

(a)Install a firewall service virtual appliance as a hypervisor module. This appliance uses the virtual machine introspection API of the hypervisor [11] and hence has visibility into all traffic flowing inside the virtual network, including traffic that never leaves the virtualized host. Specifically, this appliance sets up a firewall filter to intercept all traffic flowing between a vNIC of a VM and the vSwitch and thus provides a capability to control traffic flowing into and out of every VM resident on the virtualized host.

 ISBN: 978-1-61208-249-3

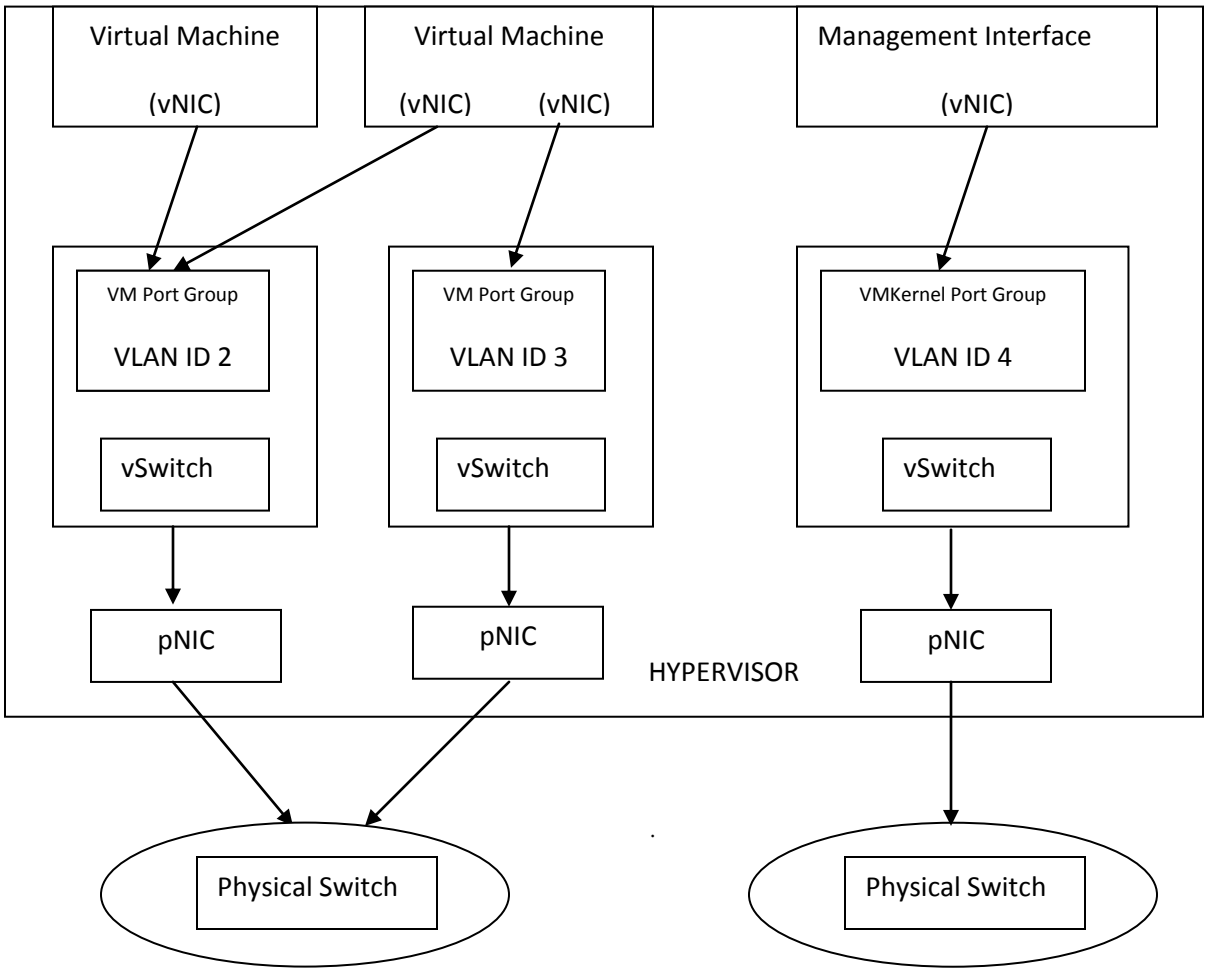

Figure 1. Typical Virtual Network Configuration

(b) Isolate the network traffic flowing among VMs using the concept of VLAN [4]. By this configuration approach, different portgroups are defined in a vSwitch and each is associated with a unique VLAN ID. A network packet originating from a VM and landing on a portgroup in a vSwitch is tagged with that associated VLAN ID. A vSwitch is connected to one or more pNICs of the virtualized host which in turn are connected to a physical switch configured as a VLAN trunk. Access control policies based on VLAN IDs are defined on this physical switch. The physical switch serves to segment traffic among the VMs resident on that host for monitoring and controlling. The difficulty with this configuration option is that the VLAN traffic inside each virtualized host must be routed to the external physical switch where access control policies are enforced on inter-VLAN traffic.

The Security requirements for the virtual network traffic isolation feature for the firewall option are:

(a)The firewall appliance should be a stateful one

(b) There must be flexibility in defining objects that will participate in policy rules. The objects should be static or dynamic. In terms of granularity of entities at the network level, objects designating portgroups and vLAN Ids should be supported. At the VM level, object definitions should cover a single designated VM, all VMs in the virtualized host or all VMs with similar connectivity or function (e.g., Web Server) [12].

(c) Traffic filtering rules by themselves cannot provide assurance against attacks on the virtual network. This requires an IDS/IPS. Since IDS/IPS systems with robust analytical capabilities are now found only in versions built for physical networks, the firewall should support rules for mirroring virtual network traffic to external network devices to leverage this capability.

 ISBN: 978-1-61208-249-3

Table 1. Hypervisor Deployment Feature (Option) & Security Assurance

Deployment Feature (Configuration Option)	Security Assurance Requirements
1. Local User Management and Authentication (using Directory Infrastructure)	No Zombie Accounts, Enforcement of Password policies, Robust Authentication Mechanism
2. CPU Scheduler Configuration (proportional fair-share)	Situations that result in Denial of Service to VMs must be minimized
3. Access Control Configuration (all options)	Arbitrary combination of Users, Permissions & Objects and Flexibility to assign Permissions at various granularity levels
4. Device Drivers Configuration (all options)	Certified Drivers, Barebones/Secure Configuration of domains running Device driver code
5. Management Interface Configuration (all options)	Dedicated Management Network, Restricting the type of Network traffic and IP address locations for Management Interface Access
6. Virtual Network Configuration (Firewall & VLAN)	Capability to define and enforce inter-VM network traffic rules and IDS/IPS function support

VIII. CONCLUSIONS & ADVANTAGES

The hypervisor is the central software that provides all of the virtualization functions on a virtualized host. Apart from the usual user account management and access control configuration options encountered on any host, a virtualized host presents a sophisticated management interface which supports functions for multiple workloads in the form of VMs and a software-defined network called virtual network. This presents a rich set of deployment features and associated with each deployment feature there are multiple configuration options. In this paper, we analyzed the security implications of these options/deployment features to derive the security assurance requirements that are either (a) provided by each of the configuration options; or (b) required for that deployment feature as a whole regardless of configuration options. The security assurance requirements are summarized in Table 1 above.

The advantages of the deployment feature-driven approach for deriving security assurance requirements are:

(a) The security assurance requirements have direct traceability to each deployment feature and hence provide an automatic test of completeness.

(b) Provides a true picture of the security posture of the operational virtualization infrastructure as the security guarantees of each deployed configuration is known in advance.

.

REFERENCES

[1] J. Sahoo, S.Mohapatra, and R.Lath, “Virtualization: A Survey On Concepts, Taxonomy And Associated Security Issues,” IEEE 2nd International Conference on Computer and Network Technology, Bangkok, Thailand, Apr 2010, pp. 222-226

[2] O.Agesen, A.Garthwaite, J. Sheldon, P.Subrahmanyam, “The Evolution of an x86 Virtual Machine Monitor”

[3] J. N. Matthews, ‘et al.’“Running Xen – A Hands-On Guide to the Art of Virtualization,” Prentice Hall, 2008

[4] S. Lowe “Mastering VMware vSphere 4,” Wiley Publishing, 2009.

[5] L. Garber, “The Challenges of Securing the Virtualized Environment”, IEEE Computer, Volume 45 Issue 1, Jan 2012.

[6] P. Colp et al, “Breaking up is Hard to Do: Security and Functionality in a Commodity Hypervisor”, ACM SOSP’11, Cascais, Portugal, Oct 23-26, 2011, pp 189-202.

[7] A. Kadav and M.W.Swift, “Understanding Modern Device Drivers,” ACM ASPLOS’12, London, England, March 3-7, 2012, pp 87-98.

[8] T. Garfinkel and M. Rosenblum, "When Virtual is harder than Real: Security Challenges in Virtual Machine Based Computing Environments" ,Stanford University Department of Computer Science. http://xenon.stanford.edu/~talg/papers/HOTOS05/virtual-harder-hotos05.pdf [Retrieved: July, 2012]

[9] Five exciting VMware networking features in vSphere 5 http://searchvmware.techtarget.com/tip/Five-exciting-VMware-networking-features-in-vSphere-5 [Retrieved: March, 2012]

[10] S. Jin, J.Ahn, S.Cha, and J.Huh, “Architectural Support for Secure Virtualization under a Vulnerable Hypervisor, “ACM MICRO’11 Conference, Porto Alegre, Brazil, Dec 2011, pp. 272-283.

[11] VMware Inc, “vShield Administration Guide – Version 5.1,” http://www.vmware.com/pdf/vshield_51_admin.pdf, pp. 9-12.

[12] Juniper Networks, Inc ,”Alternatives for Securing Virtual Networks,” http://www.juniper.net/us/en/local/pdf/whitepapers/2000382-en.pdf

 ISBN: 978-1-61208-249-3