Security Vulnerability Analysis in Virtualized Computing Environments

Tyson T. Brooks, Carlos Caicedo, Joon S. Park
Syracuse University, USA

Abstract

Virtualization brings compelling features to individual computer systems and organizations allowing for the concurrent execution of multiple operating systems and applications on the same physical server. However, for all the performance improvements offered by adopting virtualization technologies, one of the major obstacles to widespread adoption of virtualization is the concern about security in the technology. Therefore, the security implications in virtualization environments must be addressed and understood because of the exploitation to compromise the operation of mission-critical systems. In this paper, we first discuss common exploits of security properties in virtualized computing environments and analyze their security vulnerabilities from the perspective of attackers. Consequently, we identify the main areas of virtualized information system design and operation in which security concerns must be addressed. Finally, we present our recommendations and future trends for trusted virtualized computing environments.

1. Introduction

Today, virtualization technology is ubiquitously woven into nearly every technical field and conversation taking place in the world of Information Technology (IT). Alongside terms such as service-oriented architecture (SOA), Web 2.0, and cloud computing [27], virtualization is generally referred to as the technology which can deliver various benefits in terms of cost effectiveness, availability, hardware utilization, resource protection, remote access, and other capability enhancements. Previous research on virtual machine (VM) systems, which allowed software for the simulated machine to execute directly on the hardware without software interpretation [4, 15], provided groundbreaking insight into the abstraction of a computer's hardware resources from its software resources for machine virtualization. Partitioning multiple operating systems (OS) on a single processor unit identified that multiple virtual machines could share the hardware resources of a single physical machine [3]. Virtualization emulators were shown to better utilize hardware resources, resulting in increased performance with respect to timing accuracy and execution speed [13]. Virtualization gained considerable interest because it offered an opportunity to maximize the utilization of software applications, while using fewer physical devices.

Virtualization has many forms (e.g. storage, networks, OS, etc.) and the key feature is the separation of what is thought of as a logical service or system from its physical provider. The most common and one of the oldest virtualized systems is computer system memory [9]. Typically, a process may address a location address beyond the physical limitations of the actual memory installed in a computer. The key element becomes this decoupling of the logical resource from the physical resource where the current programs and data in use are held, which can affect performance. Virtualization technologies attempt to provide high virtualized system performance through open source solutions such as kernel-based VMs [20], Xen [2] and proprietary solutions such as Intel's VT-x and VT-I [54] and HyperDealer [19].

Technically, virtualization refers to the abstraction of computer resources and technology that introduces a software abstraction layer [i.e., virtual machine manager (VMM), also known as hypervisor] between the hardware and the guest OS/applications running on top of it [47]. At the highest level, virtualization is the abstraction of software logic from hardware resources. In general, abstraction is accomplished by providing a common language, which translates between different software/hardware interface constructs and instruction sets. A virtualized computing architecture provides the capability to host virtual computing environments that allows for the concurrent execution of multiple OS and applications on the same physical server, as displayed in Figure 1. Thus, it is possible to host multiple virtual computing environments, homogenously or heterogeneously, and their applications, on a single physical server platform.

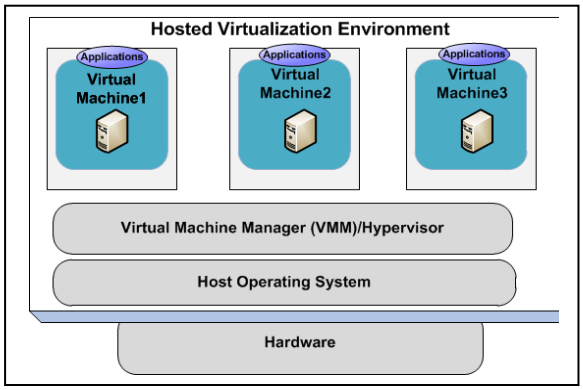

Figure 1. Virtual Computing Architecture

Overall, virtualization provides assorted advantages to service providers, organizations, and users. It allows for greater server system reliability through clustering of virtual servers with high availability features that can significantly minimize the impact of a single server failure and provides greater utilization of physical servers. These virtualized environments allow an organization to obtain greater flexibility in responding to changing demands since virtualization technologies simplify the re-provisioning and dynamic reallocation of processing capacity. However, the implications of virtual computing environments become profound and drive a shift in the fundamentals of information systems design, operation, and management. In particular, providing security in virtualized environments is a key concern and challenge in the designing and management of modern information system infrastructures.

Therefore, in this paper we first discuss common exploits of security properties in virtualized computing environments and analyze their security vulnerabilities. Consequently, we identify the main areas of virtualized information system design and operation in which security concerns must be addressed. Finally, we present our recommendations and future trends in the area.

2. Dynamics in Virtualization

Server and application virtualization represent the use of virtualization methods in information systems [28]. Server virtualization is the masking of server resources, including the number and identity of individual physical servers, processors, and OS, from server users [28]. Application virtualization abstracts the logic of the application from the logic of the operating system thereby allowing an application to run on multiple base operating system platforms absolving the requirement to develop unique driver/application programming interfaces (API) to interact with the base operating system [28].

Virtualization provides enhanced software interoperability and platform versatility between abstraction layers near hardware/software interfaces [49]. Virtualization can operate in the following modes: hosted, OS, and bare metal, as displayed in Figure 2.

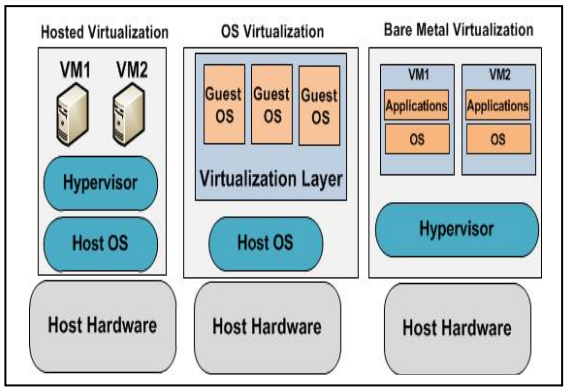

Figure 2. Virtual Computing Architecture

In a hosted virtualization environment the VMM/hypervisor, runs as a simple piece of client software within the host OS [10]. Hosted virtualization relies on a host OS and provides the interface between the VMM/hypervisor and a virtualized operating environment with a different operating system/application profile. This mode of virtualization has the highest I/O and management overhead and depends on the existing OS to provide drivers to access the physical resources [8]. OS virtualization is achieved by inserting a layer of system software between the guest OS and the underlying hardware [8]. With OS virtualization a host OS is still present and a virtualization layer is applied on top of the host OS to allow applications to run in containers which have access to resources of the host platform. OS virtualization has decreased management overhead from hosted virtualization, however it suffers from the limitation that the virtual environments must be compatible with the underlying operating system [49].

In a bare-metal environment, a hypervisor manages the operations of the virtual machines and interfaces directly with the host hardware [51]. Bare metal virtualization is based upon insertion of a hypervisor directly between the hardware resources and the OS. Therefore, it may support enhanced performance, compared with the other modes, but switching a VM requires restarting the hardware. This method of virtualization has the lowest amount of management overhead and has the flexibility of being able to run multiple OSs on a single machine. Bare metal virtualization can be implemented with virtualization logic dependent upon the kernel of the operating system, independent of kernel and

processor, or dependent upon the processor instruction set [51].

In each mode, the VMM/hypervisor is present and manages the interaction between guest software, host software, and hardware resources [2]. Table 1 compares some characteristics of these virtualization modes:

TABLE I. VIRTUALIZATION COMPARISIONS

Type	Virtualization Modes		
	Hosted	**OS**	**Bare Metal**
BaseOS	Base OS installed	Divides a single OS into containers	Lack of an existing OS
Hypervisor	Installed on OS	Runs as a distinct layer	Sits directly on top of the hardware
Access to I/O Devices	Each VM has access to limited subset of I/O devices	Host OS provides interface between VM and I/O devices	Communicates with I/O devices directly
Performance	Slow	Good; provides scalability	Fair; enables increased performance
Decoupling of Hardware and OS	Decouples the physical hardware from the OS	Decouples the physical hardware from the OS	Installs directly on hard drive; No installed OS is needed
Advantages	Users can run various Guest OS within own application	Decoupling allows for each operating system to be maintained as an image	Hypervisor emulates each shared device for guest VM; supports real-time OS
Disadvantages	Architecture not capable of providing pass-through to many I/O devices	Application require special handling from the operating system	Drivers needed to support various hardware platforms included in hypervisor
Commercial Products[a]	-VMware Workstation -VMware Fusion	-Parallels Virtuozzo -Solaris Containers	-VMware ESXi and ESX -Xen

3. Vulnerability Exploitation in Virtual Computing Environments

Attackers search for exploitable vulnerabilities in deployed virtualization environments to compromise one of its software processes (i.e. by compromising the integrity of a software process, the attacker can append malicious logic to that process). Attackers can also exploit a defect in the relationship between two components (e.g. OS, hypervisor, server, etc.). For example, by exploiting the failure of a defective component to establish a trusted path (i.e., authenticated, encrypted interface) to a second component to which it will pass sensitive data, an attacker can compromise the availability of that second component by setting up a malicious alternate component with the same internet protocol (IP) address setting the stage for a man-in-the-middle attack. Many exploits of security properties in virtualization software result from a particular sequence of steps that target a combination of defects in one or more of its components. This is especially true of components that are combined in a way that emphasizes one component's reliance on a vulnerable function or attribute of another one.

Virtual computing environments are most often compromised due to the intentional exploitation of defects that arise from inherent deficiencies in the virtual computing environment's production environment (e.g. OS level, network device and protocol level, firmware and hardware level, etc.). For example, a VMM/hypervisor main threat is the existence of potentially hostile software residing on the guest VM [47]. Hostile software could be installed as part of a commercial package within a guest VM, downloaded from the internet or embedded in email sent to the user of the guest VM. If the software running within the guest VM is untrusted, the VMM/hypervisor will have to take many safeguards to protect itself and data from hostile software [55]. Verifying trust for VMM/hypervisor software code includes the implementation of functionality such as malware reverse engineering, virus scanning, network content filters, information flow analysis and anomaly detection [4].

Detecting malicious insider activity in virtualization environments is a challenging task, which is further compounded if the insider has administrative access to the virtualization resources [6]. It may be plausible that an administrative user with high expertise in the virtual computing environment can procure the requisite amount of resources from that environment to perform denial-of-service (DoS) or man-in-the-middle attacks. The administrator user is much more likely to know about detection mechanisms and countermeasures and how to defeat them to compromise, steal, exploit or tamper with the virtual computing environment for personal gain [6]. Therefore, when assessing the robustness of a secure virtual computing environment, these types of threats should also be taken into consideration.

Failure to understand how to design and build a virtual computing environment also leads to vulnerability exploitations [21]. To determine whether the virtual computing environment is secure, adequate testing must be performed to examine its security and that of its associated development

artifacts (e.g. threat models, design specifications, etc.) to discover vulnerabilities.

While an exploitable defect will always represent a vulnerability, it is not true that all vulnerabilities in virtualization systems arise from exploitable defects. For example, consider the emergence of previously unobserved vulnerabilities that result when the virtualization software is rehosted in an environment different from that for which it was designed. These unanticipated vulnerabilities are not the result of inherent defects in the virtualization software, but instead emerge as a result of the errors in actual deployment of the virtual environment.

The motivation for producing secure virtual computing environments is the desire for these architectures to continue to operate correctly even when subjected to attack. When it comes to security, the virtualized environment not only has to be attack-resistant but also attack-resilient. This means that the environment has to be able to recover from the attack, resuming operation at or above a minimum level of service, as soon as the source of the attack has been isolated and blocked, and the damage has been contained.

4. Vulnerabilities in Virtualized Information Systems

As virtualization becomes a pervasive technology in information systems, it becomes an additional point of vulnerability from potential attackers. We will illustrate the potential vulnerabilities of virtualized information systems from the perspective of the threat community in this section.

4.1. Footprinting of Virtualized Target Systems

Successful attacks on target systems in a virtualized environment result from human ingenuity: a subtle design flaw may be exploited or a previously undiscovered implementation defect may be located through the attacker's engineering efforts. From a remote location, it is technically possible for an attacker to collect the intelligence that indicates that a target platform is operating in a virtualized environment. At a high-level, detectability is predicated upon the ability to identify anomalous patterns which exist in data streams emanating from a target network. One anomalous behavior pattern existing in virtualized environments, which is visibly apparent in external communication sessions, is the lack of virtual network time synchronization between the host OS and the guest OS [12]. The idea is that virtual time is simply counted at a different rate for the host OS and the guest OS based on their relative allocations for processing [31]. In this case, enumeration techniques could be extended to detect discontinuities between Internet Control Message Protocol (ICMP) and Transmission Control Protocol (TCP) timestamps, indicating the presence of a virtual environment [25]. This occurs because the host operating system will respond to the ICMP query via the physical network interface card (NIC), while the guest operating system will respond to the TCP solicitation via the Virtual NIC [25]. These configuration options break all communication channels between guest and host, not just for the program trying to detect the VM, but also destroying the time synchronization features of the VM.

Another technique employed by attackers is to look for discontinuities between the IP identity (ID) header values coming from the same IP address. The IP ID field is very difficult to manipulate and carries only a smidgen of useful semantic information [37]. This discontinuity indicates the presence of different number schemes for the IP ID value, which could correspond to different OS using the same network interface. These techniques allow malicious attackers to profile target systems and intuit the presence of virtualization technology in the framework of the information system. Attacks can now be tailored to queue or sequence specific tools that will take advantage of hypervisor vulnerabilities.

Locally there are many more options that allow the malicious threat to detect virtual machines. If an attacker is physically present on a network segment it is possible for it to map virtual hosts by analyzing the source media access control (MAC) address of the traffic. While a virtual machine is active, the MAC address will not change. If a source MAC address is compromised it will designate the sending node as a virtual node. In this way it may be possible to detect virtualized platforms in order to tailor attack methods. Similarly, if an attacker has compromised a machine and desires to continue the course of attack, there are several locations where the presence of artifacts or system values can be used to determine if they are executing in a real or virtualized machine environment: (1) looking for virtualized machine environment artifacts in memory, processes or system registry files (2) looking for virtualized machine environment-specific hardware and processor instructions and capabilities.

Additionally, there are several open source applications (i.e. VMDetect, ScoopyNG) which will detect the presence of a virtualized machine environment through isolated or combinations of detection methods. Once the attacker determines the presence of a virtualized machine environment, he can attempt several attacks. We discuss the actual attack methods in the following subsections. Typically, if the goal is compromise of information, the attacker could transition into a hypervisor traversal attack. If the goal is to compromise hosts for inclusion in a botnet, the attacker could move quickly to install a rootkit in the hypervisor and erase

tracks from the guest operating system. This underscores the importance of hardening the host and guest OS and providing the minimal functionality required to achieve the system's intended objectives.

4.2. Virtualized Botnets

Botnets, networks of compromised machines controlled by botmasters, have become the leading threat to Internet security [32]. Botnets can be exploited in VM environments for various purposes; the most dominant uses include distributed DoS attacks, simple mail transfer protocol (SMTP) mail relays for spam, ad click fraud, etc. [34].

Using virtualization technologies, a small array of traditional multi-processor servers can simulate virtualized botnets with numerous nodes. The behavior of the virtual botnet logic can also be continually refined until it matches the desired characteristics of the operator (i.e. web server and VM) and a predefined threshold of performance. Once performance is assured, attackers are able to sell botnet access in the black market for computer services.

The resiliency of botnets is supported by methods such as rotational domain name system (DNS) [11]. In these attacks, each infected host would run server-based services such as DNS, web, and mail in a listening state. The botnet would then be able to rotate these services actively, essentially creating pivot points to execute spam floods and distributed DoS attacks. When one DNS server, acting as the root for the attack domain, would be shut down by an ISP or added to an access control list (ACL), the nodes would shift the DNS root to another affected host. Using a similar construct, malicious threats could leverage a vulnerability that allowed access to a target platform. Then a hypervisor could be installed in the background and configured with an external interface capable of accepting remote commands.

Once a hypervisor is installed on a compromised system virtual machines could be propagated using file transfer protocol (FTP) services or an installed FTP domain. If this propagation method proved successful, an affected node could serve as a receptacle to and host for virtual machines which are configured to execute malicious attacks. Virtual machines could lie dormant in the background awaiting activation by the system by accepting commands addressed to the virtual interface and bound for the virtual machine. Then, if detected, an attack, such as a DoS attack, could be pivoted to other source networks by activating a dormant VM.

4.3. Hypervisor Traversal Attacks

The hypervisor layer is the host software layer that provides the ability to run multiple OS on a single physical host system and sits between the host's physical hardware and the virtual machines [14]. The hypervisor controls the host's processor and allocates resources as required to each guest OS. Hypervisor traversal attacks involve obtaining access to a guest operating system and then leveraging this foothold to gain access to the host operating system or other guest OS sharing the same hypervisor (e.g. buffer overflow). In specific architectures, it has been demonstrated that a threat can move into the hypervisor architecture and gain visibility of all traffic traversing the internal virtual network which connects all VMs operating on the host [14].

Additionally, if rootkits have been installed in VM host operating systems these can give capabilities to an attacker to open a backdoor to the hypervisor allowing for processing of remote commands [46]. Other methods for hypervisor traversal rely on intercepting the interrupts and API calls to the host OS, walking the interrupt vector table, by enumerating VM interface subroutines and inserting an interrupt request subroutine above the existing subroutines, or just replacement of existing subroutines [16].

4.4. Virtual Code Injection Attacks

One of the most common forms of security attacks involves exploiting a vulnerability to inject malicious code into an application and then cause the injected code to be executed [17]. In a virtual code injection attack, a virtual machine with a Trojan or other malicious code could be inserted into a virtualized environment through various methods, such as:

- An attacker could compromise a guest OS, perform a hypervisor traversal to the host OS, transfer via FTP a malicious VM to the host platform, and then instantiate it.
- An attacker could execute a man-in-the-middle attack or redirect the communication between the virtualization management agent installed on a host platform and the virtual management application housed on a central server. The attacker can illicit the unique identifier for the VM and attempt to masquerade it as a legitimate VM by replacing legitimate VM images with compromised images.
- An attacker could compromise a guest OS and begin footprinting the network in search of the storage location of the managed VMs. Once this is elicited, if access controls are not in place, it may be possible for the attacker to capture VMs and send them back to the source network for constitution and profiling. Once the VM is profiled, the attacker may modify the application profile and system variables to allow for integration with the existing managed virtual environment of the target network. Then the

compromised VM is transferred back into the storage of managed VMs on the target network.
- Following from the previous attack methods, if the attacker was able to gain administrative access to the management application he could significantly modify the contents of the VM image library.

These attacks are particularly lucrative because configuration management in virtualized operational environments (as discussed in section 5.6) is lacking within most organizations. In many cases, configuration management is performed manually with rudimentary tools such as Excel spreadsheets. Therefore, the probability of detection is low and in favor of the attacker. Until virtualization focused configuration management tools are adopted across the IT industry, insertion attacks will be a highly desirable method for attackers to compromise virtualized information systems.

4.5. Other Attacks

Vulnerabilities in virtual computing environments will always be eminent. For example, VM escape is an exploit in which the attacker runs code on a VM that allows an operating system running within it to break out and interact directly with the hypervisor [5]. Such an exploit could give the attacker access to the host OS and all other VMs running on that host. VM escape is considered to be the most serious threat to virtual machine security.

Attackers are now creating VMs and aiming for collocation by performing cross-VM attacks [41]. With the collocation of VMs on the same hypervisor platform, VM environment vulnerabilities can be exposed with side channel attacks through cache, network traffic monitoring or via utilities to share files/data. For example, it is possible to map the internal cloud infrastructure, identify where a particular target VM is likely to reside, and then instantiate new VMs until one is placed co-resident with the target [41]. This then enables the attacker to mount cross-VM side-channel attacks to extract information from a target VM on the same machine [41].

One of the most feared attacks in the threat community is the breakout attack [43]. In the simplest form an attacker is able to move beyond the guest OS and open up a communications or control channel on another guest OS on the same physical server. This could allow the attacker to change the session identifier in the cookie or uniform resource locator (URL) on the guest OS side to match the identifier of another user's session on another guest OS. This provides the attacker the ability to hijack the hypervisor and gain control of the host. Also, the attacker could possibly gain access to the individual guest OS's and bypass or corrupt the validation checking to supply incorrect data or data formats to the guest OS server.

Attacks against virtualization memory leaks occur if a failure happens between allocation and freeing of a memory region in the VMM/hypervisor [24]. When a program dynamically allocates memory space for an object, array, or variable of some other type, but fails to free up the space before the program finishes executing [48], this virtualized memory leaked region can be exploited by attackers by inserting malicious code that is written to hog memory resources and cause DoS. Repeated over time, this region can cause the program to allocate all available memory (physical memory and paging file space), with the result that all software processes on the allocating program's host suspend operation until the allocating program releases the memory. Virtualization memory leaks provide an easy entry point for buffer overflow attacks [7, 23].

In general, the emergence of vulnerabilities in virtual computing environments often result from previously undetected defects in the host/guest OS or VMM/hypervisor. However, they may also be the result of novel and ingenious exploitations of the virtualization technology by an attacker attempting to achieve malicious results. This said, strategies for secure virtualization solutions should be developed to anticipate any defect-related vulnerabilities.

5. Components of Virtualized Environment Security

In general, implementation of virtualization technology can improve numerous aspects of IT services. However, there are also new security concerns that must be addressed to maintain the existing security posture. Conceptually, the tenets or foundations of information security remain unchanged with the introduction of virtualization. Practically, virtualization can have a broad impact, affecting the current methods, controls, and processes associated with information systems.

While previous work has been performed on identifying virtualization security [22], we recognize seven areas of virtualization security, discussed below, which require special consideration in virtual computing environments: application security, storage security, hypervisor security, host/guest security, network security management, operational security and guest OS security.

5.1. Application Security

In application virtualization, the user is able to run a server application locally, using the local resources without needing the complexity of completely installing the application on his/her computer [44]. Application security includes various

technologies designed for encapsulating user mode applications and isolating them from the underlying OS [44]. In essence, each application is encased in a bubble that prevents the application from modifying the host file system and registry. Although application virtualization is generally intended as a means for improving compatibility, portability, and manageability of applications, security is often touted as a benefit. If applications are truly encapsulated, then compromising an application cannot result in a compromise of the host.

Application virtualization is not the same as OS virtualization. The former virtualizes the runtime environment of a single application. The latter virtualizes the computing environment of an entire OS. The implementations of application virtualization can vary dramatically. For example, when an application is virtualized, it can be hosted on a central file server. Instead of installing the application on each client system, the application can be streamed to clients on demand. This reduces the attack footprint on the client because fewer applications are natively installed. In addition, patch management is improved because only the file server has to be updated.

Application security through virtualization involves a different paradigm than typical OS-based security. A great deal of work has examined how to protect security-critical portions of applications from the OS through mechanisms such as microkernels, VMMs, and new processor architectures [38]. For example, using OS based security functions, malware could be denied write-access to system folders or registry keys because it runs with lower rights than those required to perform the corresponding actions. However if the system is protected by using application virtualization, the write-access would be allowed, but would modify only the encapsulated environment, which does not affect the host machine. The security benefits that application virtualization provide are only as good as the integrity of where each VM is isolated from its host physical system and other VMs.

5.2. Storage Security

Storage requirements for virtual environments greatly exceed those of the traditional operational environments due to additional OSs, applications, and virtual machine images that must be maintained as a part of the processing and management architecture [18]. A diverse collection of storage products allow for logical storage volumes created across arrays of physical storage resources. While an in-depth analysis of storage technologies is beyond the scope of this article, there are several recommendations to consider for secure storage virtualization. All baseline images and stateful snapshots of VMs should be stored and managed locally to accelerate timetables for reconstitution in the event of a disaster or hardware failure. Accordingly, logical volumes should not span physical arrays, which contain disparate information domains. Essentially, information domains should be logically and physically separated. Periodic snapshots of virtual images should be taken and backed up to a remote location to mitigate the impact of a regional disaster event. In addition, storage resources should be maintained by dedicated administrative staff and access should be granted only to authorized administrators.

More recently, storage virtualization has become a mainstream concept. Networked file systems previously provided an elementary form of secure virtualization in data centers [26, 40]. Today, storage virtualization provides a complete abstraction of logical storage from physical storage [39]. The physical storage units are combined by the virtualization device into storage pools from which the logical storage resources are created. The logical storage is presented to the user for data storage while the mapping to the physical storage is transparently handled by the virtualization device. By separating the physical storage from the application, maintenance activities and data migration can be performed without any impact to operations. Storage virtualization also provides an environment that can scale to meet future storage needs and allows for a secure migration of data.

5.3 Hypervisor Security

The hypervisor is likely to become the target of more focused attacks and, as a result, will become more resilient. Since the hypervisor sits between the host OS and guest systems, it is imperative to ensure this component is as secure as possible. Any defects in hypervisor security may impact both the host and guest systems. The job of the hypervisor is to translate between host and guest hardware instruction sets [26]. The thinner the hypervisor and more closely integrated with the processor instruction set, the smaller the opportunity for compromise due to vulnerability. Using virtualization-enabled hardware products, the hypervisor can be configured to prevent the execution of code from memory locations reserved for data storage. This is typically an attack vector used during buffer overflow attacks. Buffer overflow exploits are targeted at web application and web service components that accept data as input and store it in memory (rather than on disk) for later use or manipulation [26]. An overflow of a memory buffer results when the web application or service component fails to adequately check the size of the input data to ensure that it is not larger than the memory buffer allocated to receive it, and instead passes the too-large data into the too-small buffer

[7]. As hypervisor capabilities continue to evolve, it is recommended that implementations move towards thin hypervisor architectures that are stored in Non-Volatile Random Access Memory (RAM) and can perform a trusted boot sequence.

Some VMs that communicate between VMs on the same hypervisor may pass through the VMM/hypervisor unencrypted. This is similar to the way two computers communicate across a local area network (LAN) in a traditional environment. Just as in the traditional environment, if someone gains unauthorized access to the communications equipment between two systems they may observe the data, except in the virtualized machine environment, the equipment is the VMM/hypervisor. Therefore, it is essential that security controls for both the hypervisor and host are operational.

5.4 Host/Guest Security

The security relationship between hosted hypervisors (e.g. a guest system and the host system) in a virtualized environment is much the same as the relationship between a computer system and application in the traditional environment. With a traditional computer system almost any vulnerability that is applicable to the computer itself will somehow impact the applications executing on the system. This still holds true in the virtualized environment as well; almost any vulnerability on the host hypervisor will impact the security posture of the guest systems. For example, the configuration files for constituting and executing the guest systems are typically stored on the host system. If these files become compromised they inherently compromise the guest system. Also, in host and OS virtualization, the hypervisor is stored on the host OS.

In bare metal environments, there are many explicit and implicit interfaces between untrusted guest software and the hypervisor [50]. Some of these include executing privileged instructions, interacting with emulated devices and using virtualization-specific services like a dynamic host configuration protocol (DHCP) server on a virtualized network. Each virtual computing environment exposes an interface that resembles real hardware to its VM, thereby giving the host/guest OS the illusion of running on a bare-metal platform [50]. Nevertheless, bare metal virtualization adds potentially vulnerable privileged interfaces to its host/guest environments and provides enhanced performance for VMs which helps to limit potential exploits.

Furthermore, attackers could potentially find flaws in host/guest virtualized environments. Virtual machine detection could become a significant security risk as a precursor to virtualized machine environment escape—a procedure in which malicious code running inside a guest machine can escape and begin running on the host [14, 16]. Attackers are highly motivated to move from the guest system to the host system because this provides the means to affect additional services and access additional data resources.

5.5 Network Security Management

In comparison to traditional architectures, virtual network systems which leverage virtualization can be constituted from a fractional amount of physical resources [1]. The processing power, memory, and storage resources are allocated dynamically with security controls potentially changing at the same time. If a VM becomes overloaded, peers within the same virtual environment can be temporarily shifted to another server. Within the virtualized environments, the standard device-level enumeration of the physical system no longer applies. Instead a meta-architecture now applies in that the physical systems are identified along with virtual devices (including processing, memory, and storage) as they are instantiated. If security controls are not identified and evaluated across architectural layers, true defense-in-depth cannot be accomplished which may result in security controls not working as expected.

A key architectural issue in virtualized environments is the location and placement of the management console. The management console can use the control features of the layer to observe, stop, reboot, load-balance, and otherwise manage the server with greater ease than if they were running on physical boxes [44]. Just as system administration activities should be performed with a different user account and administration consoles should be located on separate networks, the management station for the virtualized environment should be located outside of the virtualized environment of the managed machines. It is recommended that network management communications occur via a separate physical network interface. This out-of-band management channel should be isolated so that it can be monitored in depth. Traffic between the management node and the production environment should be explicitly denied, with the exception of approved management protocols. Wherever possible, stateful inspection techniques can be applied to management channels in an attempt to identify anomalous behavior and activities with a potential for system compromise.

Furthermore, virtualization allows multiple VMs to share the resources of a single physical machine, including the physical network interface, which allows multiple OS to safely coexist [44]. If multiple VMs are sharing a single Ethernet interface, each VM can have its own IP address or the VMs can share a single IP address through Network Address Translation (NAT) performed by the virtualized

Network Interface Card (NIC). This provides a certain degree of protection occlusion from intrusion detection system (IDS)/intrusion prevention systems (IPS) devices that are deployed on the LAN. Traffic which flows between VMs on a single box is essentially invisible to these devices. In most live environments, core services will require static IP addresses, so there may be a one-to-one mapping of virtual interfaces to physical interfaces thereby alleviating this problem.

5.6 Operational Security

In traditional environments adding or removing storage, new computers, and networking devices required physical machines that could be tracked throughout the change control process (purchasing, inventory, staging, etc.). In virtualized environments, these systems and devices may be added or removed with a few simple clicks. Many network monitoring and management tools are unable to gain meaningful visibility into the inner workings of virtual networks, as well as falling short in their ability to storage, track, and analyze changes within the virtual environment. Additionally, the ability to quickly take system "snapshots" and move back and forth between system configurations adds new concerns for tracking and managing patches/updates, licensing, and access controls (e.g., one snapshot has different user accounts than another). Virtualization provides new ways to obfuscate attack vectors and subvert traditional detection techniques. Through NAT, it is possible to share an IP address among many virtual machines preventing analysis by IDS/IPS devices. VMs behind NAT do not have end-to-end connectivity and cannot participate in some Internet protocols which result in many applications relying on publicly accessible servers for dissemination of data between NAT VMs [52]. To mitigate these subversion techniques rigorous configuration management, access control [33], and auditing controls must be implemented.

Configuration management is important to operational security since it involves identifying the configuration of a system at given points in time. By systematically controlling changes to the configuration, and maintaining the integrity and traceability of the configuration throughout the whole lifecycle, configuration management is one of the most important and practical disciplines necessary for effective delivery of IT services. In relation to operational security, configuration management is the cornerstone of information security providing the backdrop to audit the system baselines in search of anomalies that could indicate the occurrence of an information attack. Decoupling between the hardware and software creates more motion among system elements thereby emphasizing the need to implement rigorous configuration management practices. Because IT administrators and managers can flexibly build out new services, as well as expand existing services almost instantly, rigorous configuration management practices must be in place to ensure that licensing compliance, vulnerability patching, and integrity of the system components are maintained. The ability to take snapshots of VM configurations is extremely desirable from the point of view of service continuity, troubleshooting, and integration/testing of new applications. However, the management of machine state adds considerable overhead and can quickly become a confusing maze thus increasing the chances of propagating an unpatched system into the production environment.

Through rigorous configuration management practices coupled with virtualization technology, disaster recovery procedures are drastically improved. If a service suffers a catastrophic failure or even a minor discontinuity, the service can be reconstituted immediately by restoring a virtual machine from a catalogue or library of VM images. The machine in question can be shut down and moved into a separate network segment for further analysis. At the same time, a machine with an approved configuration in a secure state can be migrated onto the hardware framework without impact to existing services. Immediately the machine can be turned on and integrated into the live environment, but only if there is a high degree of confidence that the machine will not introduce vulnerabilities into the system.

5.7 Guest OS Security

In the virtualized environment it is easy to think of each guest as an island unto itself. The key security consideration for guest OSs is that it has a direct interface to the hypervisor, which can act as a path to the host OS. Additionally, all VMs on a physical machine are interconnected via the virtual NIC and therefore represent a logical network segment [52].

Currently there is a lack of product capability to provide internal monitoring services to these virtualized networks. For these reasons, it is imperative that guest OSs and the applications running in guest OSs are locked down according to industry best practices and configuration guidance. With multiple logical machines consolidated on a single physical machine, as discussed in Section 4, any vulnerability in the guest OS becomes a vulnerability to all of the processing services operating on the physical machine. The security practice of minimal functionality should be employed liberally to ensure that VMs are only providing essential functionality and accessibility to support desired services.

6. Technical Countermeasures Against Virtualized Attacks

Countermeasures that are aimed at specific threats and vulnerabilities involve more sophisticated techniques as well as activities that are traditionally perceived as security protocols. In order to mitigate and reduce the types of attacks mentioned in previous sections, appropriate countermeasures must be employed based on their effectiveness in mitigating the vulnerabilities of virtualization computing environments.

Countermeasures for virtualization environments should take into consideration the probability of data loss and criticality. Technical countermeasures are preventive controls which stop attempts to breach system security. As part of the computer hardware, software or firmware (e.g. access control mechanisms, encryption, etc.), they are intended to determine if the vulnerabilities can be mitigated through practical countermeasures such as audit trails and intrusion detection systems which warn administrators of violations or attempted violations of system security. Countermeasures for virtualized environments are designed to reflect an understanding of security principles (e.g., least privilege, simplicity). These countermeasures can take the form of security wrappers (e.g. filtering/blocking input/output), application-level firewalls (e.g., filtering/blocking, I/O), environment constraints (e.g., sandboxes, trusted OS access controls), compiler security extensions (e.g., open source components), and code signing (e.g., mobile code components).

Table I lists typical attack implications to virtualized computing environments and summarizes a set of recommended countermeasures for each.

TABLE II. VIRTUALIZED ATTACK COUNTERMEASURES

Virtualized Attack Implications	Countermeasures
Footprinting of Virtualized Target Systems	Security wrappers, application-level firewalls: Countermeasures for address spoofing require correctly configuring perimeter security. Rejecting incoming packets from the Internet that contain an internal ("behind the firewall") IP address in their header, as well as rejecting outgoing packets when their headers indicate that the packets originated from an external IP address.
Virtualized Botnets	Security wrappers, application-level firewalls: Techniques for detecting and handling botnets can be applied towards DNS attacks. In some ways, attacks against DNS/web servers are easier to detect than those against other applications (e.g. database), because web service payload information is more readily available. With the right tools, message traffic patterns indicating possible DoS attacks can be detected even when the same or similar payload is being sent via multiple communications protocols, e.g., FTP, SMTP, or across different physical or logical interfaces.
Hypervisor Traversal Attacks	Environment constraints, compiler security extensions: By allocating only non-executable storage areas for input buffers, any attack code embedded in oversized inputs will not be inadvertently executed. This approach can be used to stop those buffer overflow attacks that have the objective of executing malicious code, but will not counteract buffer overflow DoS attacks.
Virtual Code Injection Attacks	Code signing; secure library versions: Structured query language (SQL) injection attacks are most effectively prevented by applying thorough application layer countermeasures, such as web application firewalls, and better yet, by explicitly designing and implementing the web service logic added to legacy database applications to resist/reject all input that contains SQL injection attack patterns. Network-level firewalls and intrusion detection systems, and database security controls, have not proved effective in defending against SQL injection attacks.

7. Recommendations

The ability to determine the security of networking and OS virtualization technology is an inexact science at best. Security measures for protecting OS, networking software, and to a lesser extent middleware (e.g., database management systems, web servers) are proving effective in resisting many attack patterns. Due to its ubiquity and increasing criticality, attackers are recognizing that virtualization technology has greatly increased in value as a target. Users trust and rely on secure software for every business function they perform as well as a number of non-business functions. Because virtualization technologies and their vulnerabilities have become so familiar, attackers are gaining ever-increasing knowledge of the often widely-reported exploitable defects (e.g. vulnerabilities) in commonly-used software technologies, and thus are able to craft increasingly effective attacks against software that incorporates those technologies.

The list of attack implications enumerated in section 6 is not provided in order to indicate that any single solution would be more, or less, secure than another. The listing is intended to identify vulnerabilities to indicate that although the technology may be relatively new to an enterprise environment, the amount of implications evident in the products is growing as use increases. To mitigate the risk associated with rogue or phantom virtualization architectures, as well as unpatched or improperly patched VMs, the following are several security recommendations aimed at securing these types of environments:

- Establish a common nomenclature to name VM images. Ensure that the version and patch date are included as part of the nomenclature.
- Store all VM images in a single logical storage location, or VM image library. Ensure that the VM library shares an internal connection to the production environment. All VM images should be protected by integrity controls such as checksums. Checksum information should be stored separately from the VM image library. Logically separate the storage of active VMs and archived VMs. Active VMs should identically mirror the corresponding live configuration of the operational information system.
- Logically separate the storage of active VMs and archived VMs. Active VMs should mirror the corresponding live configuration of the operational information system.
- Document all approved VM configurations maintained as a part of the VM image library. Ensure documentation is updated with every version change to the approved VM configuration.
- Allocate dedicated administrative resources to the role of Virtualization Administrator. This is not a collateral duty. This is frequently an overlooked cost of virtualization, however, the dimension of complexity as well as instability associated with emergent technologies clearly drive the need for dedicated administrative resources.
- Protect logical storage of active VMs with access controls. Read access should be limited to specific administrative roles, but should be denied to standard system users [20]. Deny all write, copy, and execute access to this directory, except for an administrative role associated with virtualization management.
- Perform high-level audit of active VMs on a weekly basis. Ensure integrity of the structure and nomenclature. Once a month perform spot testing or sampling of active VMs but constituting a randomly selected, active VM in a closed testing environment, then scanning the VM for patch, license, architectural, and configuration compliance.

8. Future Trends

The majority of virtualization attacks are preventable by applying the necessary level of due diligence with early phases of the development lifecycle (e.g., requirements, architecture, design, and so on) for implementing virtualization technologies. However, no matter how faithful a security-enhanced virtualization architecture is adhered to, as long as virtualization technology continues to grow in size and complexity, a number of exploitable virtualization defects can be guaranteed to appear. Because poor programming practices or simple human fallibility will enable errors with security repercussions to be introduced during implementation or install-time configuration, the design of a virtualization technology cannot be deemed vulnerability-free.

One of the most promising future elements of virtualization technology is the ability to rapidly expand the capacity of network services through the activation and porting of VMs into a live operational environment. In the past, if an information system was designed to proxy and scan Internet data requests, it was limited by the physical machine's memory size and architecture. In order to satisfy increased demand above the physical limitations of hardware, additional memory would have to be installed or a new machine would have to be built. In virtual computing architectures, hardware resources are shared and processing demand, imposed by software, is balanced across these resources. If the virtual computing architecture is capable of providing additional hardware resources, increasing the capacity is a matter of moving a virtual machine from a storage network into the operational network.

Once the device is activated and configured to share the load of the proxy services, the performance bottleneck is alleviated. This of course assumes that the resource constraint is not external bandwidth, but rather processing and storage resources of the information system. The degree of dynamic scalability is a welcome advance from the point of view of network administrators and IT professionals. However, it is also a welcome advance from the perspective of the attacker. The scalability factor increases the ease of executing DoS attacks once a target is enumerated and a vulnerability discovered. This means that the attacker's chances of gaining access or disrupting additional systems are increased from the availability of access to these additional devices. In the future, virtualization will enable much more insidious attacks on external interfaces, such as routers, gateway devices, proxies, firewalls, etc., as well as legitimate services which are not designed to manage large traffic flows and exponential expansion of data requests. However, it is possible to block DoS attacks at gateways and network boundaries.

These days the growth of cloud computing will provide experience on how to effectively and securely share common infrastructure among mutually untrusting entities [35; 42, 53]. Cloud computing is becoming an enabler of the global computing virtualization concept and has proven to be an incredible technology for provision of quickly deployed and scalable IT solutions at reduced infrastructure costs. Likewise, virtualization is also a future computing infrastructure which organizations are looking to reduce cost, improve service levels, and increase operational flexibility. Infrastructure as a Service (IAAS), a major feature of cloud

computing, is the use of virtualization to abstract infrastructure so users can deploy VMs over a choice of physical hardware and data centers. Using cloud computing across virtualized computing infrastructures could potentially enhance access and delivery of applications and data to the end users at a rapid rate. Unfortunately, with these positives the use of cloud and virtualization technologies raise larger issues related to security, latency, inadequate service levels, governance, data protection, maturity and reliability. These are the issues which prevent cloud-computing solutions from fully utilizing virtualization and becoming the prevalent alternative for critical IT systems.

Recently, the virtual security services architecture (VSSA) approach introduced the concept of a security layer that logically and securely cordon's off virtualization servers from the rest of the corporate network [30]. Specifically, this layer is used to allow a remote display protocol to be passed between servers and the client machines, with all the necessary security software provisioned for each corporate workstation at once. The security control domain manager provides an SOA like approach in identifying from the security services registry for the creation and management of secure VM's. Before a VM is implemented in a production environment or a major security update is needed to the existing infrastructure, security services are requested and provisioned from the security services processor. The VMM/hypervisor provides the virtual operating platform to monitor the security control domain and guest OS's. The VMs are where the actual client work will take place and the remote display software is used to provide remote display of the VMs to corporate workstations. These workstations display information and the corporate network provides the network path between the workstations and virtualization servers. The corporate SAN provides the data repository for storing data and audit logs. Rather than provide security separately for each VM, the security layer simultaneously provides this protection for all VM's in use throughout the organization.

As technology improves, business drivers such as improved security and cost savings will push organizations and businesses to adopt virtualization technology. The primary driver, improved security, focuses upon improvements in the security posture of the information system through enhancements in the availability of critical services. From the perspective of the processing infrastructure, the increase in availability contributes to an increase in the overall availability of the information system, which can translate into improvements in the operational efficiency of the business or organization. The secondary driver, cost savings, increases the Return on Investment (ROI) for major IT expenditures. Cost savings for virtualization is realized through three major reductions: reduction in physical servers, reduction in rack space, and reduction in energy consumption. Asset reduction has an immediate and tangible effect of decreasing procurement costs and labor costs associated with activities such as installation and life-cycle maintenance. In addition, reduction in energy consumption decreases the recurring power and heat management costs associated with operations and maintenance of information systems.

Behind each of these drivers, cost could be potentially weighed against the captured value realized by a movement to virtualization technologies. As with any technology change, there are significant investment costs, which offset the potential costs savings. For all of the promised cost savings, many early adopters are finding hidden costs that accumulate and slowly erode the analysis, which supported the business decision to embrace virtualization technology.

9. Conclusions

Although virtualization provides assorted advantages to service providers, organizations, and users, it also introduces new challenges and concerns related to implementing secure virtualized computing environments. Therefore, in this paper, we have discussed common exploits for these environments from the perspective of threat community and identified the main components of virtualization security as well as a set of countermeasures to secure these environments. Finally, we have presented our recommendations and future trends in the area.

Virtualization technology is being rapidly assimilated into IT architectures. It is clear that organizations are adopting virtualization in an effort to drive down life-cycle IT costs. Regardless of the virtualization product, secure virtualization enables dynamic and agile delivery of services within an organization construct. In theory, this means that applications will be securely delivered to the user desktop on-demand, driven by real-time business needs and businesses are free from paying license costs for unused applications.

10. References

[1] Ahuja, M. and Carley, K. (1999) 'Network Structure in Virtual Organizations', *Organization Science* 10 (6), pp. 741–757.

[2] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., and Warfield, A. (2003) 'Xen and the Art of Virtualization', in proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP'03), ACM: New York, NY, vol. 35, pp. 164-177.

[3] Borden, T., Hennessy, J. and Rymarczyk, J. (1989) 'Multiple Operating Systems on One Processor Complex', *IBM Systems Journal* 28 (1), pp. 104-123.

[4] Bratus, S., Locasto, M., Ramaswamy, A. and Smith, S. (2010) 'VM-based Security Overkill: A Lament for Applied Systems Security Research', in proceedings of the 2010 New Security Paradigms Workshop (NSPW), ACM: New York, NY, pp. 51-60.

[5] Chunlei, W., Wen, Y., and Yiqi, D. (2010). A Software Vulnerability Analysis Environment Based on Virtualization Technology. in proceedings of the 2010 IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS), IEEE Computer Society Press: Los Alamitos, CA, pp. 620-624.

[6] Claycomb, W. and Shin, D. (2010) 'Detecting Insider Activity using Enhanced Directory Virtualization', in proceedings of the 2010 ACM Workshop on Insider Threats, ACM: New York, NY, pp. 29-36.

[7] Cowan, C., Wagle, P., Pu, C., Beattie, S., and Walpole, J. (2000) 'Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade', in proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'00), IEEE Computer Society Press: Los Alamitos, CA, vol.2, pp. 119-129.

[8] Crosby, S. and Brown, D. (2006) 'The Virtualization Reality', *Queue* 4 (10), pp. 34-41.

[9] Creasy, R. (1981) 'The Origin of the VM/370 Time-Sharing System', *IBM Journal of Research and Development* 25 (5), pp. 483-490.

[10] Davy, S., Fahy, C., Griffin, L., Boudjemil, Z., Berl, A., Fischer, A., Meer, H., and Strassner, J. (2008) 'Towards a Policy-Based Autonomic Virtual Network to Support Differentiated Security Services', in proceedings of the International Conference on Telecommunication and Multimedia (TEMU), Ierapetra: Crete, Greece, pp. 1-7.

[11] Dittrich, D., Leder, F., and Werner, T. (2010) 'A Case Study in Ethical Decision Making Regarding Remote Mitigation of Botnets', in *Financial Cryptography and Data Security*, R. Sion, R. Curtmola, S. Dietrich, A. Kiayias, J. Miret, K. Sako and F. Sebé, Eds. Springer: New York, NY, pp. 216-230.

[12] Etsion, Y., Ben-Nun, T., and Feitelson, D. (2009) 'A Global Scheduling Framework for Virtualization Environments', in proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Processing (IPDPS 2009), IEEE Computer Society Press: Los Alamitos, CA, pp. 1-8.

[13] Falcón, A., Faraboschi, P., and Ortega, D. (2007) 'Combining Simulation and Virtualization Through Dynamic Sampling', in proceedings of the IEEE International Symposium on Performance Analysis of Systems & Software (ISPASS 2007), IEEE Computer Society Press: Los Alamitos, CA, pp. 72-83.

[14] Garfinkel, T., Adams, K., Warfield, A., and Franklin, J. (2007) 'Compatibility is Not Transparency: Vmm detection myths and realities', in proceedings of the 11th Workshop on Hot Topics in Operating Systems (HOTOS XI), USENIX Association: Berkeley, CA, article no. 6.

[15] Goldberg, R. (1974) 'Survey of Virtual Machine Research', *IEEE Computer Magazine* 7 (6), pp. 34-45.

[16] Grimes, R. (2007) 'Excellent VM Detection and Breakout Presentation' InfoWorld; http://www.infoworld.com/d/security-central/excellent-vm-detection-and-breakout-presentation-333 (15 March 2011).

[17] Hu, W., Hiser, J., Williams, D., Filipi, A., Davidson, J., Evans, D., Knight, J., Nguyen-Tuong, A. and Rowanhill, J. (2006) 'Secure and Practical Defense Against Code-Injection Attacks Using Software Dynamic Translation', in proceedings of the 2nd International Conference on Virtual Execution Environments (VEE '06), ACM: New York, NY, pp. 2-12.

[18] Huang, L., Peng, G., and Tzi-cker, C. (2004) 'Multi-Dimensional Storage Virtualization', in proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '04/Performance '04), ACM: New York, NY, vol. 32, pp. 14-24.

[19] Hwang, W., Roh, Y., Park, Y., Park, K. and Park, K. (2010) 'HyperDealer: Reference-Pattern-Aware Instant Memory Balancing for Consolidated Virtual Machines', in proceedings of the IEEE 3rd International Conference on Cloud Computing (CLOUD), IEEE Computer Society Press: Los Alamitos, CA, pp. 426-434.

[20] Ibrahim, K., Hofmeyr, S. and Iancu, C. (2011) 'Characterizing the Performance of Parallel Applications on Multi-Socket Virtual Machines', in proceedings of the 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE Computer Society Press: Los Alamitos, CA, pp. 1-12.

[21] Kotsovinos, E. (2010) 'Virtualization: Blessing or Curse?' *Queue* 8(11), pp. 1-6.

[22] Kroeker, K. (2009) 'The Evolution of Virtualization', *Communications of the ACM* 52 (3), pp. 18-20.

[23] Larochelle, D. and Evans, D. (2001) 'Statically Detecting Likely Buffer Overflow Vulnerabilities' in proceedings of the 10th conference on USENIX Security Symposium (SSYM'01), USENIX: Berkeley, CA, pp. 177-190.

[24] Le, M. and Tamir, Y. (2011) 'Enabling VM Survival Across Hypervisor Failures', in proceedings of the 7th ACM SIGPLAN/SIGOPS Internal Conference on Virtual Execution Environments (VEE '11), ACM: New York, NY, vol. 46, pp. 63-74.

[25] Liston, T. and Skoudis, E. (2006) 'On the Cutting Edge: Thwarting Virtual Machine Detection', Intelguardians; http://handlers.sans.org/tliston/ThwartingVMDetection_Liston_Skoudis.pdf (1 March 2011).

[26] McGregor, T. and Cleary, J. (1998) 'A Block-Based Network File System', in proceedings of the 21st Australasian Computer Science Conference (ACSC'98), Springer: New York, NY, pp. 133-144.

[27] Mell, P. and Grance, T. (2011) 'The NIST Definition of Cloud Computing', Recommendations of the National Institute of Standards and Technology, Special Publication 800-145, January 2011.

[28] Menascé, D. (2005) 'Virtualization: Concepts, Applications, and Performance Modeling', in proceedings of the 31st International Conference for the Resource Management and Performance Evaluation of Enterprise Computing Systems (CMG2005), The Computer Measurement Group: Turnersville, NJ, pp. 407– 414.

[30] Naqvi, E. (2010) 'Designing Efficient Security Services Infrastructure for Virtualization Oriented Architectures', in Pervasive Information Security and Privacy Developments: Trends and Advancements, H. Nemati, Ed. Information Science Reference, Hershey, PA, pp. 149-171.

[31] Nieh, J. Vaill, C. and Zhong, H. (2001) 'Virtual-Time Round Robin: An 0(1) Proportional Share Scheduler', in Proceedings of the 2001 USENIX Annual Technical Conference, USENIX: Berkeley, CA, pp. 245-260.

[32] Ormerod, T., Lingyu, W., Debbabi, M., Youssef, A., Binsalleeh, H., Boukhtouta, A., and Sinha, P. (2010) 'Defaming Botnet Toolkits: A Bottom-Up Approach to Mitigating the Threat', in proceedings of the Fourth International Conference on Emerging Security Information, Systems and Technologies (SECUREWARE 2010), IEEE Computer Society Press: Los Alamitos, CA, pp. 195-200.

[33] Park, J., An, G. and Liu, I. (2010) 'Active Access Control (AAC) with Fine-Granularity and Scalability', *Security and Communication Networks* 4(10), pp. 1114-1129.

[34] Park, J., Lu, H. and Tsui, C. (2009) 'Anti-Spam Approaches: Analyses and Comparisons', *The Open Information Systems Journal* 3, pp. 36-47.

[35] Park, J. and Robinson, J. (2010) 'Trusted Content-Sharing Services in Cloud Computing', in proceedings of the International Conference on Cloud Computing & Virtualization (CCV 2010), Global Science & Technology Forum: Singapore, Singapore, pp.188-194.

[36] Park, J., Sandhu, R. and Ahn, G. (2001) 'Role-Based Access Control on the Web', ACM Transactions on Information and System Security (TISSEC) 4 (1), pp. 37–71.

[37] Paxon, V. (2001) 'An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks', *ACM SIGCOMM Computer Communication Review* 31(3), pp. 38-47.

[38] Ports, D. and Garfinkel, T. (2008) 'Towards Application Security on Untrusted Operating Systems', in proceedings of the 3rd USENIX Workshop on Hot Topics in Security (HOTSEC '08), USENIX: Berkeley, CA, article no. 1.

[39] Qiang, Z., Yunlong, W., Dong, C. and Zhuang, D. (2010) 'Research on the Security of Storage Virtualization Based on Trusted Computing', in proceedings of the 2010 2nd International Conference on Networking and Digital Society (ICNDS), IEEE Computer Society Press: Los Alamitos, CA, vol.2, pp. 237-240.

[40] Rajasekar, A., Wan, W., Moore, R., Schroeder, W., Kremenek, G., Jagatheesan, A., Cowart, C., Zhu, B., Chen, S. and Olschanowsky, R. (2003) 'Storage Resource Broker – Managing Distributed Data in a Grid', *Computer Society of India Journal* 33 (4), pp. 42–54.

[41] Ristenpart, T., Tromer, E., Shacham, H. and Savage, S. (2009) 'Hey, You, Get Off of my Cloud: Exploring Information Leakage in Third-Party Compute Clouds', in proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09), ACM: New York, NY, pp. 199-212.

[42] Robinson, J. and Park, J. (2010) 'Towards Trusted Cloud Computing', in proceedings of the 2010 iConference, iSchool iConference: Urbana-Champaign, IL, pp. 188-193.

[43] Roschke, S., Cheng, F. and Meinel, C. (2009) 'Intrusion Detection in the Cloud', in proceedings of the Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC '09), IEEE Computer Society Press: Los Alamitos, CA, pp. 729-734.

[44] Rosenblum, M. and Garfinkel, T. (2005) 'Virtual Machine Monitors: Current Technology and Future Trends', IEEE Transactions on Computers 38(5), pp. 39-47.

[45] Rosenblum, R. (2004) 'The Reincarnation of Virtual Machines', *ACM Queue* 2(5), pp. 34–40.

[46] Rutkowska, J. (2006) 'Subverting Vista Kernel for Fun and Profit', Black Hat Briefings; http://blackhat.com/presentations/bh-usa-06/BH-US-06-Rutkowska.pdf (1 March 2011).

[47] Sahoo, J., Mohapatra, S., and Lath, R. (2010) 'Virtualization: A Survey on Concepts, Taxonomy and Associated Security Issues', in proceedings of the Second International Conference on Computer and Network Technology (ICCNT), IEEE Computer Society Press: Los Alamitos, CA, pp. 222-226.

[48] Silva, L., Alonso, J. and Torres, J. (2009) 'Using Virtualization to Improve Software Rejuvenation', *IEEE Transactions on Computers* 58(11), pp. 1525-1538.

[49] Smith, J. and Nair, R. (2005) 'The Architecture of Virtual Machines', *IEEE Computer* 38(5), pp. 32–38.

[50] Steinburg, U. and Kauer, B. (2010) 'NOVA: A Microhypervisor-Based Secure Virtualization Architecture', in proceedings of the 5th European Conference on Computer Systems (EuroSys '10), ACM: New York, NY, pp. 209-222.

[51] Stelte, B., Koch, R. and Ullmann, M. (2010) 'Towards Integrity Measurement in Virtualized Environment- A Hypervisor Based Sensory Integrity Measurement Architecture (SIMA)', in proceedings of the 2010 IEEE International Conference on Technologies for Homeland Security (HST), IEEE Computer Society Press: Los Alamitos, CA, pp. 106-112.

[52] Tang, Y. and Li, J. (2009) 'Towards Overlay Network in Virtual Computing Environment', in proceedings of the International Conference on Apperceiving Computing and Intelligence Analysis (ICACIA 2009), IEEE Computer Society Press: Los Alamitos, CA, pp. 274-277.

[53] Treglia, J. and Park, J. (2009) 'Towards Trusted Intelligence Information Sharing', in proceedings of ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics (CSI-KDD '09), ACM: New York, NY, pp. 45-52.

[54] Uhlig, R., Neiger, G., Rodgers, D., Santoni, A., Martins, F., Anderson, A., Bennett, S., Kagi, A., Leung, F. and Smith, L. (2005) 'Intel Virtualization Technology', *IEEE Computer* 38(5), pp. 48-56.

[55] Vaughan-Nichols, S. (2008) 'Virtualization Sparks Security Concerns', *Computer* 41(8), pp. 13-15.

