© 2014 IJIRT | Volume 1 Issue 5 | ISSN : 2349-6002

VIRTUAL MACHINE SECURITY

Sonarka Maini, Saurabh Jakhmola
Information Technology
Dronacharya College Of Engineering

***Abstract*- Computer security is a chronic and growing problem, even for Linux, as evidenced by the seemingly endless stream of software security vulnerabilities. Security research has produced numerous access control mechanisms that help improve system security; however, there is little concensus on the best solution. Virtualization plays a major role in helping the organizations to reduce the operational cost, and still ensuring improved efficiency, better utilization and flexibility of existing hardware. "Virtualization is both an opportunity and a threat -says Patrick Lin, Senior director of Product Management for VMware". This paper presents a literature study on various security issues in virtualization technologies. Our study focus mainly on some open security vulnerabilities that virtualization brings to the environment. We concentrate on security issues that are unique for virtual machines. The security threats presented here are common to all the virtualization technologies available in the market, they are not specific to a single virtualization technology. We provide an overview of various virtualization technologies available in the market at the first place together with some security benefits that comes together with virtualization. Finally we provide a detailed discussion of several security holes in the virtualized environment.**

Index Terms- Virtualization, Security ,Operating System

I. INTRODUCTION

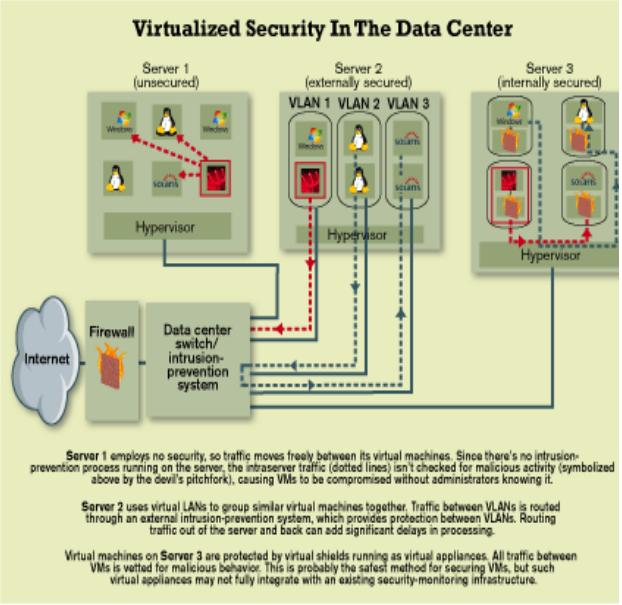
Security is a chronic and growing problem: as more systems (and more money) go on line, the motivation to attack rises. Linux is not immune to this threat: the "many eyes make shallow bugs" argument not withstanding, Linux systems do experience a large number of software vulnerabilities.

Security is a very broad concept, and so is the security of a system. All too often, people believe that a system is way more secure that it in practice is, but the biggest problems is still the human factor of the users; the possibility of careless or malicious users are commonly overlooked.

Virtualization essentially introduces a level of indirection to a system to decouple applications from the underlying host system. This decoupling can be leveraged to provide important properties such as isolation and mobility, providing a myriad of useful benefits. These benefits include supporting server consolidation by isolating applications from one another while sharing the same machine, improved system security by isolating vulnerable applications from other mission critical applications running on the same machine, fault resilience by migrating applications of faulty hosts, dynamic load balancing by migrating applications to less
loaded hosts, and improved service availability and administration by migrating applications before host maintenance so that they can continue to run with minimal downtime.
In non-virtual environment, the applications running on the machine can see each other, and in some cases can even communicate with each other, whereas in virtual environment the programs running in one guest machine are isolated from the programs running in another guest machine, in other words guest machines "provide what appear to be independent coexisting computers" to their running programs. The degree of isolation should be strong enough that the vulnerabilities in one virtual machine should not affect either the virtual machines or the underlying host machine.

OS virtualization provides a _ne granularity of control at the level of individual processes or applications, which is more beneficial than the hardware virtualization abstraction that works with entire OS instances. For example, OS virtualization can enable transparent migration of individual applications, not just migration of entire OS instances.This granularity migration provides greater exibility and results in lower overhead [21, 24].

Furthermore, if the operating system requires maintenance, OS virtualization can be used to migrate the critical applications to another running operating system instance. By decoupling applications from the OS instance, OS virtualization enables the underlying OS to be patched and updated in a timely manner with minimal impact on the availability of application services [26]. Hardware virtualization alone cannot provide this functionality since it ties applications to an OS instance,and commodity operating systems inevitably incur downtime due to necessary maintenance and security updates. On the other hand new security protection programs are also emerging in the market every now and then from different vendors, but most of these security solutions are mainly focused on hypervisor. Since hypervisor is a new layer between the host's OS and virtual environment, it creates new opportunities for the malicious programs. And more over, hypervisor is basically a software program, so it has all the traditional software bugs and the security vulnerabilities as any software have. One of such product that hits the market recently is SHype [4], a new secure hypervisor that binds security policies to the virtual environment. A good debate on recent security solutions can be found on [10].



However, virtual machine security is more than just deploying a secure hypervisor to the environment. Virtualization technologies are still evolving. Newer versions with added features are introduced before the security consequences of the older version has been fully studied. This work analyzes the general security threats in a virtual environment and suggests possible solutions for few of the mentioned threats. Understanding of virtualization technologies greatly helps to understand the security consequences that occur in the environment. The back ground of various virtualization technologies together with some security benefits offered by these virtualization technologies and finally analyze the security issues concerning virtualization.

II. RESEARCH METHODOLOGY

This paper is a literature survey that analyse various issues concerning security in virtual machine environment. This work provides an overview of security consequences arises in a virtualized environment. However this paper does not provide one perfect solution for all the described threats. But do provide an understanding of how these threats can be avoided while implementing virtualization.

III. VIRTUALIZTION CONCEPTS

OS virtualization isolates processes within a virtual execution environment by monitoring their interaction with the underlying OS instance. Similar to hardware virtualization [25], applications that run within the virtual environment should exhibit an e_ect identical to that demonstrated
as if they had been run on the unvirtualized system. In addition, a statistically dominant subset of the application interaction with system resources should be direct to minimize overhead.
We classify OS virtualization approaches along two dimensions, host-independence and completeness. Host-dependent virtualization only isolates processes while host-independent
virtualization also decouples them. The distinction is that host-dependent virtualization simply blocks or filters out the namespace between processes, while host-independent virtualization provides a private virtual namespace for the applications' referenced OS resources. The former does not support transparent application migration since the lack of resource translation tables mandates that the resource

identifiers of an application remain static across hosts for a mi-
grating process, which can lead to identifier conicts when migrating between hosts. Examples of host-dependent virtualization include Linux VServers [34] and Solaris Zones [27].

Host-independent virtualization encapsulates processes in a private namespace that translates resource identifiers from any host to the private identifiers expected by the migrating application. Examples of this approach include Zap [21, 24] and Capsules [30]. We refer to this virtual private namespace as a pod, based on the terminology used in Zap.In terms of completeness, partial virtualization virtualizes only a subset of OS resources. The most common example of this is virtual memory, which provides each process with its own private memory namespace but doesn't virtualize any other OS resources. As another example, the FreeBSD

IV. CONCLUSIONS

SE Linux provides a much more fine-grained control over the security of a Linux system compared to the “Unix" standard. The baseline security configuration is almost usable for most environments, but some configuration is needed in most cases. The difficulty of configuration has maybe been the reason why most people have not taken SE Linux in use, but the policy management tools are getting better.

Some people claim4 that the security framework provided by LSM is not extensive enough, that several critical security hooks are missing and that SE Linux security relies on the kernel being bug free. These claims are probably at least partially true, but the latest development in SE Linux tries to address the remaining security issues (e.g. minimize the processed that can change the policy etc). Complete system security is an utopia, but SE Linux is one step in that direction. It is being included in most major Linux distributions, even though it might not be enabled by default. Installing or activating SE Linux is pretty straight-forward, and no enforcement is being done until the user has checked the log files for possible problems and decides that the configuration is good enough.

REFERENCES

[1] Linux Security Modules Paper
[2] Mandatory Acess Control Wiki
[3] Smack Security Module Wiki
[4] LSM Project
[5] LINUX Security Modules Wiki
[6] Kernel Security Documentation