COEN 329 Project:
Virtualization Applications

by

David Obatake
W0782384

Santa Clara University
Santa Clara, California
November 15, 2014

Audience

This report is for people interested in learning about what virtualization is and what the purpose of virtualization is in today's networked-personal computing environment. The report will cover what virtualization is in its many forms, as well as the benefits of virtualization in consumer and enterprise settings. The reader should have general knowledge of operating systems and networks to understand the basics of computers, users, and their needs in networked computer infrastructures.

Table of Contents

Table of Figures

Chapter 1

Introduction

The primary focus of this report is virtualization: its implementations, motivations, and applications. Differentiating between various types of virtualization show two main types of virtualization: platform and resource virtualization. Platform virtualization focuses on obfuscating the underlying hardware and providing a simulation of computing environments for easier management and encapsulation of hardware resources. On the other hand, resource virtualization aggregates or encapsulates various computer resources in some simulated form that provides simplified access to storage, memory, or network resources. There are many applications of virtualization available, each utilizing process abstraction and/or encapsulation to provide a more manageable and understandable system for both users and administrators. Further breaking down the categories of platform and resource virtualization allows for a more thorough understanding of virtualization overall and the benefits/challenges of the uses of virtualization applications.

In chapter 2, the report will first describe virtualization in a broad definition and also look at the common aspects of virtualization across platform and resource virtualization. This chapter will cover full virtualization, partial virtualization, and paravirtualization. Then the report will move onto a discussion of platform virtualization applications, discussing server virtualization, application virtualization, desktop virtualization, and operating system-level virtualization. The next chapter will go into resource virtualization such as storage and network virtualization. The section on network virtualization will look at VPNs, NATs, VLANs, VPLS. Chapter 5 will cover the future aspects of virtualization looking at new innovations in the field of virtualization and close the report in a short summary.

Chapter 2

What is Virtualization?

In the modern computing environment, the meaning of virtualization changes depending on the context of the application or technology. Many references to virtualization on the Internet are used in relation to the enterprise IT industry where virtualization is used to reduce overhead costs of equipment. Still, virtualization is a broad topic that covers both computer hardware and resources that can mean the use of virtual hardware or resources that simulates the physical hardware or resources in some way. The virtual simulation of the physical infrastructure allows for separation of implementation from how the system is used. This allows for a substitution of the actual hardware and resources for virtual resources that abstract the technical details away so that the end user of the system does not generally know about the substitution of the virtual resources while the system still performs in the same way.

Originally, virtualization was implemented to provide timesharing for users on IBM OS/360 computers through a virtual machine monitor (VMM). This VMM separated multiprogramming and the extended machine aspects of an operating system that allowed for more than one user at a time. The VMM ran on the bare hardware of the computer and provided multiple exact copies of the hardware that included the kernel, I/O, interrupts, and everything else in a computer. Each copy of the hardware was a virtual machine (VM) that provided a convenient interface for multiple users which could have their own operating system (OS) that worked independently from each other. Each VM could host its own OS, even different operating systems from the host OS that would normally take multiple sets hardware and installations. In this way, each user could have their own space of operation

without interfering with other users on the same system. This form of virtualization is called full hardware virtualization, where the VMM simulates a full set of hardware for each VM running on top of the VMM. Completely separating the hardware interface from the system running on top, allowed each part to be simpler and more flexible, resulting in a more maintainable system that could be shared among multiple users while reducing overhead costs from investing and maintaining multiple sets of hardware.

Since then, the field of virtualization has expanded to include a multitude of different technologies with the underlying principle of separating or abstracting the use from the underlying resources as the main concept. The purpose of virtualization is to organize the principal parts of the computer or resource to make them simpler to manage and more efficient to use. We can divide the field of virtualization into two main camps, platform and resource virtualization. Platform virtualization covers the different modes of simulation of computer environments, which are full, partial, and para-virtualization. Platform virtualization has several types of applications that utilize the different modes of platform in varying capacities and they are: server, desktop, OS-level, and application virtualization. On the other hand, resource virtualization describes the simulation of specific combined or simplified resources that include storage, network, memory, and CPU core virtualization.

Many individuals confuse virtualization with the concept of cloud computing and use those two concepts interchangeably. While it may be the case that cloud computing utilizes virtualization, in many aspects virtualization is a more specific term denoting system architecture and resource management that is used to provide a better computing experience to its end users. Server, storage, and application virtualization are some of the many types of virtualization implemented on remote servers to provide a cohesive working environment to the many users that access data centers to offset the workload on local devices. Thus, cloud computing is a more general term used to refer to the many types of virtualization used on remote servers that allow access to more resources and hardware via network connection.

Full Virtualization

Full hardware virtualization is a mode of platform virtualization that simulates all hardware of the host machine that runs software to allow for multiple guest operating systems to run

within its own container or VM. Each VM is independently run and managed on top of a VMM so that each system can run as if it were installed on its own hardware platform. This means that different operating systems can be run concurrently on the same set of hardware through multiple VMs running on a single operating system. Each guest OS could run unmodified as all hardware on the host system was virtualized or simulated for each VM running on the host system. Resources of the host machine: networking, storage, CPU, and memory are pooled together and delivered as needed to each VM by a piece of software called the hypervisor or VMM.

Motivation

Most motivation behind full virtualization is a product of operational efficiency and infrastructure consolidation. By allowing multiple VMs to run in parallel on one set of hardware, more workload maximizes the use of the host hardware. Instead of having to purchase a workstation level computers for each employee where it may not be used all of the time, one server can host multiple employees throughout the day and dynamically allocate resources and computing time to the current users. Also, when there are more intensive or time-sensitive jobs, the VMs with those jobs can be prioritized and request more resources of the host system, which may not be available on a single workstation, to expedite task completion.

Implementation

Full virtualization is implemented by virtualizing or simulating all hardware of the host computer system including the BIOS, I/O, and resources. By simulating all of the hardware resources, fully virtualized systems can run unmodified OS without access or porting of the guest OS kernel. Full virtualization is achieved in three ways: binary translation, software emulation, or hardware-assisted virtualization.

The first way of using full virtualization combines binary translation and direct execution of OS commands. Some non-virtualizable (or unsafe) instructions in the kernel are translated into a new sequences of instructions that have the desired effect on the simulated hardware. Otherwise, the instructions are directly executed on the virtual CPU of the VM.

Through this method, each guest OS has full access to the underlying hardware as long as those resources are available thus it does not need to be modified for specific use within a VM. The guest OS instructions are translated as needed and cached for future use, while user applications run unmodified at native speed.

Emulation may also be used to accomplish full virtualization by translating every instruction including user applications and kernel code. This makes operation slow for guest operating systems on an emulated system, but the draw to emulation remains that it allows for code written for one type of hardware to be run on different hardware (ex. run 32-bit code on 64-bit machine or vice versa).

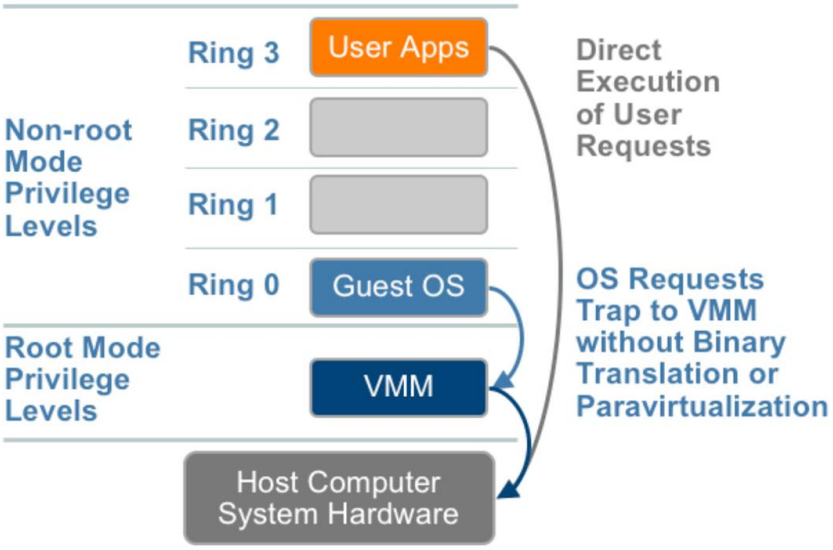

Figure 2.1: Hardware-Assisted Virtualization Architectural Model

The third way to implement full virtualization is through hardware assisted virtualization that target privileged instructions. These instructions used for specific OS kernel and resource management are usually non-virtualizable, but with the use of special CPU hardware, namely Intel Virtualization Technology or AMD's AMD-V, those instructions are trapped to the hypervisor without the need for binary translation or paravirtualization. The hypervisor is able to run in a special root mode that stores the guest state which further facilitates VM entry and exits. Figure 2.1 shows the architectural model of hardware

assisted virtualization with the VMM in root mode.

Hypervisors

There are two types of hypervisors that are implemented through either software or firmware. A type 1 hypervisor is usually referred to as a bare-metal or firmware installation that runs directly on the hardware and is installed first as the operating system of the host machine. This type of hypervisor directly interacts with the underlying hardware by virtualizing and delivering the resources needed for each VM. Type 2 hypervisors are installed on top of an existing OS, where the OS provides the abstraction of host resources and then the hypervisor delivers those resources to each VM. Although the resources are directed through another layer, type 2 hypervisors do not introduce much latency and can be used optimally in the correct computing environments. The figure 2.2 visualizes the difference between the the two types. However, not all hypervisors are equal; different types of hypervisors are desirable for various applications and workloads that the hypervisor will manage. Metrics such as maturity, ease of deployment, manageability, automation, support, performance, scalability, security, reliability, and availability are all areas that need to examined in finding the right hypervisor for each use case.

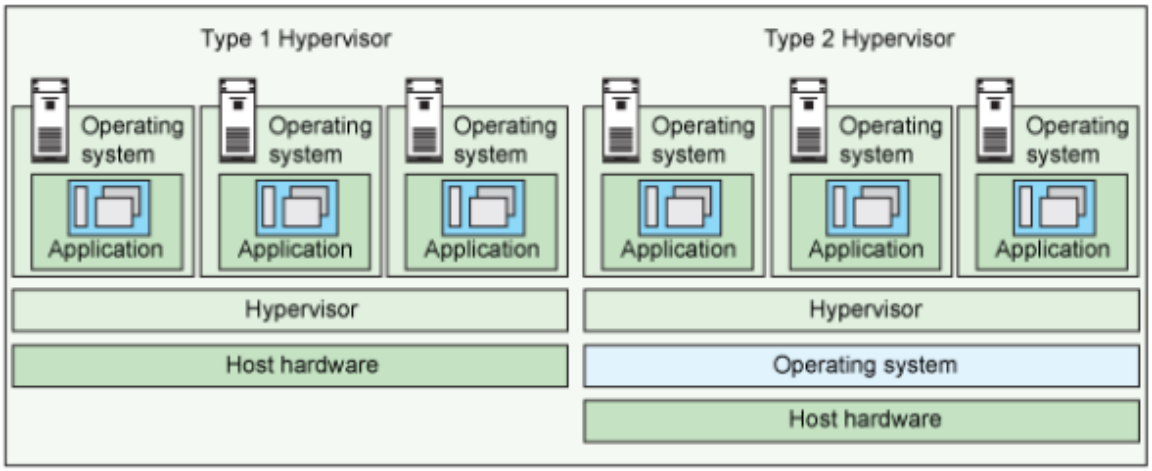

Figure 2.2: Type 1 and 2 Hypervisors

These two kinds of hypervisors divide full virtualization into two types: native and hosted full virtualization. Native virtualization results from using a type 1 hypervisor and hosted virtualization from a type 2 hypervisor. On hosted system, the OS of the VM can

even be different from the host OS without modification of the guest OS. Usually there is a virtualization application running on the guest OS that allow the user to interact with the host OS by controlling the virtual environment and sharing files with the host OS. Hosted systems also allow the user to run other applications alongside the virtual instance, while native virtualization only allows the user to run the virtualized application.

Advantages and Disadvantages

Security

Security issues with full virtualization resolve to mostly the underlying OS or hypervisor that run on the hardware itself. Because the VMs are isolated from the network via the underlying virtualization software and from one another, most attacks on a single VM can be isolated to that VM and can be contained there. For type 1 hypervisor virtualizations, the hypervisor itself is the vulnerability of the whole system, and if taken over by malware can compromise each VM on the system. In hosted virtualizations, the host OS is the main vulnerability of the system as with a regular computer environment. The main difference between the two types of implementations is the number and size of vulnerabilities present. Generally a hypervisor on the bare metal is smaller target with less security vulnerabilities as compared to a regular OS if the hypervisor is well secured. Also, type 1 hypervisors are generally compatible with only a limited amount of the available hardware, as they must be able to communicate and instruct the underlying hardware without further instruction translation. However, a hypervisor running on a host OS provides more functionality for the end user while also increasing complexity and vulnerabilities on the system because of the host OS.

Portability

Encapsulating and decoupling each VM from the underlying hardware isolates each VM from the host system that it operates on, thus allowing the VM to run on different sets of hardware without having to modify the OS each time a guest needs to be migrated to a new system. In enterprise settings, portability is a crucial metric to review because when one set of hardware inevitably fails each OS should be re-installed on a new set of hardware

to make that system available to either the employees or customers. When the hardware goes down, the cost of downtime goes up due to maintenance and loss of business, thus when a VM is portable it greatly reduces downtime because the VMs may be migrated to another set of hardware without having to re-install each guest OS on a separate set of hardware. Instead, only the host system needs to be reconfigured on a different set of hardware while the crucial VMs may be transferred to a working set of hardware until the new set of hardware is ready for operation. While portability might not be a major concern for the end user, server administrators may need to ensure that their system is available all of the time.

Also, many fully virtualized systems provide recoverability through snapshots, which are saved states of the VM at a certain point in time. These snapshots are easier to implement and faster to use than full backup or restores.

Applications

In the security field, full virtualization is used to fully simulate computer environments so that researchers can examine the effects of computer malware while being able to control the operating environment. Full virtualization provides the necessary means to create and maintain multiple OS instances on the same machine that operate as if the guest systems were installed on their own hardware. The hypervisor can then be configured and controlled to allow or block the malware from executing by providing different resources to examine the effects the the malware as if it were infected on a regular computer. These tests can be conducted manually or through an automated process without having to re-install the OS every test run. However, it is still important to secure the VM and hypervisor from malware attacks in general, as new malware is adapting to security measures constantly and are able to detect whether or not they are in a VM to adapt their attacks.

In a way, full hardware virtualization allows opens the door for the rest of platform virtualization. In some sense server, OS-level, desktop, and application virtualization all depend on full virtualization to simulate multiple sets of hardware that allow those types of virtualization to operate in parallel on the same host system. Full virtualization is only one type of virtualization available to these applications and allows for specific benefits of

running unmodified operating systems so that the guest system does not realize that the underlying hardware and resources are delivered via a hypervisor.

Partial Virtualization

Partial virtualization is another mode of platform virtualization that simulates multiple instances of only some underlying hardware of the host platform. For example, the address spaces of the host system are virtualized to support resources sharing and process isolation. Partial virtualization is a type 2 hypervisor or hosted hypervisor that runs on a host OS. Because only some of the hardware is virtualized for the guest OS, the other hardware is accessed via the host OS. However, partial virtualization does not support full hardware simulation so guest operating systems cannot be supported via a VM without modifying the guest OS. As with full virtualization, partial virtualization can be used to share computer resources and services between several computer users.

Motivation

From the patent filed for partial virtualization, the inventors reference the need for a mode of virtualization that allows a guest OS running within a virtual machine on top of a hypervisor, to be aware of its virtual environment so that the guest OS can make direct requests to the underlying hardware without having to go through the hypervisor. The reasoning behind the implementation of partial virtualization is to reduce the latency in full virtualization where the hardware requests made in the guest OS must undergo double processing first by the guest OS, then by the hypervisor, and then the same processing must happen for the hardware response when returning to the guest OS. So, in partial virtualization the guest OS may take several paths in requesting resources from the host system: through the hypervisor, to an actual driver on the host system, or directly to the host hardware. In some circumstances, the number of steps or processing time is reduced which may affect functions or programs that require high speed or data flow.

Implementation

Because not all of the hardware is simulated in partial virtualization, partial virtualization is somewhat easier to implement on the host side in software in comparison to full virtualization. There are a variety of applications of that utilize partial virtualization to some degree that only require simulation of one particular resource, thus only requiring implementation of that resource for the guest OS.

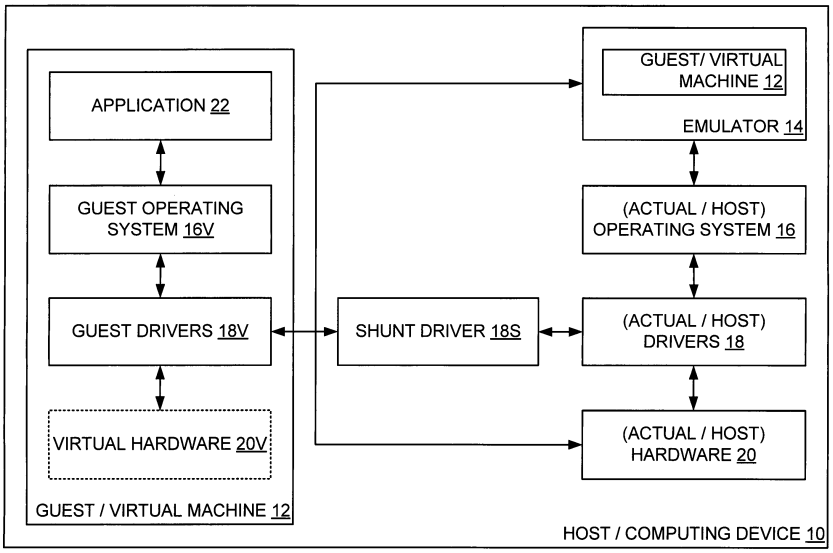

Figure 2.3: Partial Virtualization on a Host Machine

Figure 2.3 shows how partial virtualization would be implemented on a host machine with hardware requests bypassing the hypervisor to go directly to the host hardware or make requests to the hardware drivers. This model allows for faster processing for hardware requests the guest OS in the virtual machine may make during run time.

Advantages and Disadvantages

Security

Partial virtualization supports tighter models of security by limiting the scope of malware that may execute on the guest OS. However, if there are unsafe paths either directly to the underlying hardware or OS, or indirectly through the hypervisor to access resources outside the VM, then vulnerabilities exist where malware may break out of the virtualized

environment.

Portability

The disadvantages of having a partially virtualized system include specifically modifying the guest OS specifically for the host system/hardware and making sure that the guest OS is compatible with the host hardware. The modifications to the guest OS include the changes to allow it to recognize that it is in a virtual machine as well as add or modify instructions to allow the guest OS to make direct requests to the host OS or hardware. Modifying the guest OS for partial virtualization may lead to backward compatibility or portability issues for the guest OS. Also, if certain applications require particular hardware features, those applications may run into operational issues if certain portions of the host hardware are not simulated or appropriately modified in the guest OS.

Applications

The applications of partial virtualization include address space virtualization in memory, where each process has its own private memory space but the applications that utilize this virtual memory space do not receive any other virtualized OS resources. In BSD jails, partial virtualization provides restricted access to the filesystem, network, and processes outside of the jail, but does not regulate the interprocess communication (IPC) mechanisms.

Paravirtualization

Paravirtualization is a mode of hardware virtualization that does not simulate any hardware but instead provides an application programming interface (API) that can be integrated into a guest OS for operation on the host machine. VMware describes paravirtualization as an "alongside virtualization" where the guest OS communicates with the hypervisor to improve overall efficiency and performance. Whereas the guest OS in a fully virtualized system does not know that it running within a VM, paravirtualization provides a front-end for the guest OS to make calls to the paravirtualized backend on the host machine. The guest OS is modified so that non-virtualizable instructions in the kernel are translated into hypercalls to the virtualization level hypervisor. This provides the guest OS with kernel

operations such as memory management, and IPC. In comparison to partial virtualization, paravirtualization only allows for the guest OS to make hardware requests through the API that the paravirtualization software provides, where partial virtualization provides multiple methods for the guest OS to make requests to the hardware drivers or hardware itself on the host machine.

Motivation

The value trade from compatibility to virtualization overhead implementation is the main factor behind paravirtualization. Lower virtualization overhead means that paravirtualization can greatly increase performance of the VMs running on the host computer, specifically for disk and network operations where paravirtualized device drivers have near native performance. Lower overhead also results in a simplification of the VMM through the API specific calls. While the API may not be as robust as a full virtualization VMM, the complexity of the code needed to implement paravirtualization is much less due to the restrictive nature of the API in comparison to the binary translation or emulation of the computer system in full virtualization. The resulting downside of paravirtualization is a lack of compatibility with unmodified operating systems because the only way for the guest OS to communicate with the virtualization layer is through the paravirtualization API. This also means that support and maintenance for paravirtualized systems can be issues in business environments because of the deep kernel modifications that are implemented to make operating systems compatible with the host computer and the virtualization API.

Implementation

As mentioned before, paravirtualization uses an interface that translates kernel instructions into hypercalls made to the virtualization layer where the hypervisor is able to deliver hardware resources to the paravirtualized kernel. Figure 2.4 shows the architectural model of paravirtualization, where the guest OS incorporates a paravirtualized front-end that communicates to the host system's paravirtualization backend. Here, the kernel is able to determine at run time whether it is running under virtualization which will prompt it to use optimized low-level instructions for the specific virtualization stack. Other virtualizable

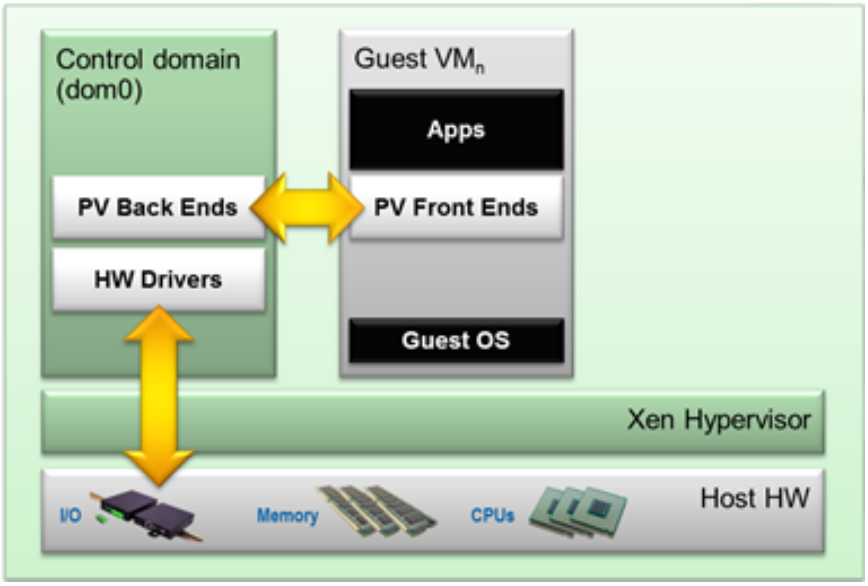

Figure 2.4: Xen's Model of Paravirtualization

instructions will execute directly on the host hardware without going through paravirtualized API. The control domain exists as the extension of the available hardware in the form or drivers and API calls that the paravirtualization software presents to the guest OS.

Paravirtualization avoids much of the overhead of full virtualization by not virtualizing many of the host hardware features by changing the non-virtualizable instructions through modifications to the guest kernel. Also, paravirtualization does not have to emulate or translate kernel code, so the code size and complexity of the virtualization layer or hypervisor is reduced and easier to verify in its operation. Modifications to the guest kernel means that portability to other systems or other VMMs is reduced and supporting the paravirtualized system after a hardware refresh may mean more downtime.

Paravirtualization may be used in combination with hardware-assisted virtualization to provide access to virtualized hardware via paravirtualized driver components. This method of virtualization is called hybrid virtualization and utilizes the raw hardware provided on the host computer via the hardware-assisted virtualization but also addresses practical problems such as VM traps and CPU overhead through paravirtualization as to reduce the amount of processing between the root mode VMM and the underlying hardware by providing access to drivers through the paravirtualized API.

Chapter 3

Platform Virtualization Applications

Server Virtualization

Server virtualization relies on hardware virtualization, whether it be full virtualization, partial virtualization, or paravirtualization, to obfuscate the underlying hardware resources like number and identity of individual servers, processors, and VMs that may be running in a data center or server room. Through hardware virtualization, servers can be partitioned into multiple isolated server instances that can serve a number of users at any given time. Server virtualization takes full advantage of full virtualization solutions such as running multiple operating systems in isolation from one another, even running different operating systems for different applications that may be needed by the users. Each guest system does not know that it is running on the same system as other guests because of the full hardware simulation presented by each VM and the resources delivered by the hypervisor.

Server virtualization is simply an extension of hardware virtualization but instead of running on a desktop or workstation computer, server and rack computers are virtualized and then accessed over a network connection. This allows for computer resource consolidation via server racks where more hardware infrastructure can be built out to server many users at once. The core of server virtualization is an implementation of an advanced load-balancer where servers are able to dynamically manage themselves based on the activity or workload on the system without input or management by server administrators. Usually the server provides access to a number of services or applications for multiple end users. On

the backend, a proxy will load-balance the workload of the users accessing the server with a number of different servers, distributing the computation and pooling resources among those servers. To the outside world there is only one server that handles the user requests thus securing the server by only allowing one entry point to the server and its applications. Server virtualization provides more robust services and comprehensive availability by load balancing the requests to multiple servers, so even in the case of one failure other servers are able to handle user requests.

Many of the other applications of hardware virtualization such as desktop, OS-level, and application virtualization mentioned in this report are subsets of server virtualization, just as server virtualization is a subset of hardware virtualization. However, it is not just the hardware that is virtualized in server computing; resource virtualization is also incorporated into the server virtualization structure. Storage and network virtualization are also key components in providing a better user experience in virtualizing servers. Offloading resources and hardware to the data centers means that the clients that access the data and servers do not need to be a full fledged computer, rather they can be thin clients that simply show the results or data to the end user.

Advantages

In the IT field, server virtualization is a solution for many enterprise and small businesses who wish to reduce overhead costs of equipment, deployment, support, and security while also balancing the operational management of their hardware infrastructure against client scalability. Enabling server virtualization allows businesses to invest in server infrastructure and management to allow for greater network traffic and system availability. VMs can be migrated between servers without downtime and VM snapshots provide VM state backups for quick recovery. Even if a single VM crashes, that VM will not bring down other VMs running on the same system. Consolidation allows for greater scalability through virtualization without increasing the cost per worker needed for computer infrastructure.

When needed, hardware and software refreshes only need to be done on the server hardware, which means less downtime for workers and more time for the IT staff to handle other problems. Refreshes for the servers mean that software updates and patches are also

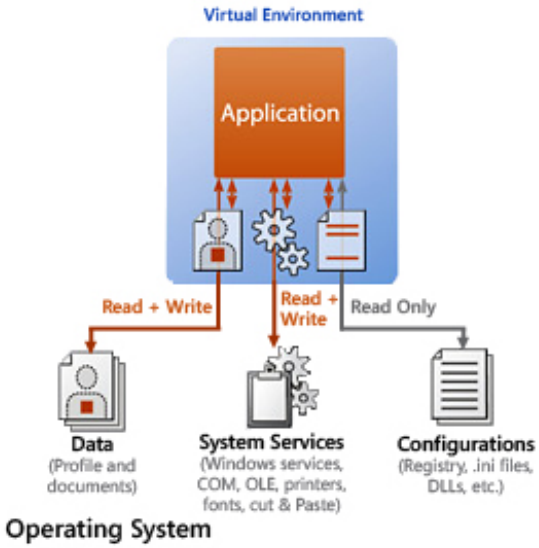

Figure 3.1: Application Virtualization Model

provided for remote clients without the need for installation on the client themselves.

Application Virtualization

Virtualizing applications involves removing or encapsulating the application from the operating system and hardware platform. Separating applications from the platform allows those applications to be packaged and ported to multiple devices without specific requirements of hardware or OS. Applications themselves are encapsulated on their own so they do not have to share address spaces with other applications, thus removing compatibility issues with other applications or operating systems. Virtualized applications are not installed on the end user's device, instead they are hosted or streamed from a remote server and displayed on the device via a remote display protocol over a network connection. Even without installation, the virtual applications can be integrated into existing desktops to appear and behave like local applications. Figure 3.1 shows the data interchange between the thin client and the remote server where the application is hosted or stored.

Hosted/Remote Applications

Hosted applications require a constant network connection to the hosting server but these applications are run completely on the server itself. The application itself can be shared on

the server among multiple users as with a terminal services desktop or the application can run on its OS instance for one user.

Streamed Applications

Streamed applications require that the client device execute the application in some capacity using either CPU processing or memory, or both. When the application is needed, the server dynamically sends the needed parts to the client for operation. As with a regular program on a local machine, not all of the program needs to be in memory to operate, only needed parts of the program are present so that memory can be conserved. Once completely downloaded, the network connection is no longer needed unless the client device cannot hold all of the program data within memory at once. Using partial virtualization, applications can be isolated within their own address spaces without having to share client device resources and interfere with other applications on the local machine.

Advantages and Disadvantages

Both hosted and streamed applications hold the benefits of server virtualization. Consolidation of software to a server means that the application can be installed, patched, upgraded, and distributed via the central data center that hosts the application. However, the challenge with application virtualization is the number of software licenses required for authorization of use which may increase capital and operational expenditures for businesses looking to implement application virtualization. Both the application software and the virtualization software must be licensed correctly before the end user can take advantage of this model of virtualization.

Applications

Microsoft's App-V software is one of many application virtualization solutions that allow users to access their applications on any authorized device without having to install those applications multiple times. Microsoft also provides a Remote Desktop protocol that streams applications to client devices straight from the cloud.

Citrix also provides an application virtualization solution via its XenApp software. Citrix differentiates its streaming and hosted applications as application virtualization and

session virtualization respectively. In its session virtualization, XenApp connects the user to the hosting server where the application is being executed, then by remotely sending input to that server the user can interact with their application.

Desktop Virtualization

Conceptually, desktop virtualization is the process of removing the operating system instance from the physical computer or client used to access the OS. In many cases, data centers or remote servers will host the OS using virtualization to provide the necessary hardware resources to each of the OS instances. In recent years, desktop virtualization has been called virtual desktop infrastructure (originally coined by VMware), specifically to mean that there is a VM for each user running on a server. These solutions like Microsoft's Hyper-V and Citrix's XenServer hypervisors provide this architectural framework that allows servers to run multiple VM instances to host multiple users. While it may cost more than a simple remote desktop access application, VDIs provide a better user experience and greater manageability and availability. In order to talk about the specifics of we can split desktop virtualization into two general categories: client-based processing or hosted processing.

Hosted Virtual Desktops

Hosted processing utilizes server hardware to do the processing required in running the OS. The hosted model offloads the work from the client, so the client does not need to be full hardware client to use the OS or the applications. These clients can be smartphones, tablets, thin or zero clients that do not have much processing power themselves but are able to access the servers that the OS is running via a network connection. The users are then able to interact with their desktops using a remote display protocol. Figure 3.2 shows a typical deployment of a desktop virtualization server that is able to serve multiple clients that may not be able to host a full fledged OS.

In hosted desktop infrastructures, there are two types of host-based forms of desktop virtualization, namely host-based virtual machines and shared hosted desktops. Each of these types can then be used in either desktop virtualization servers that are on-site or

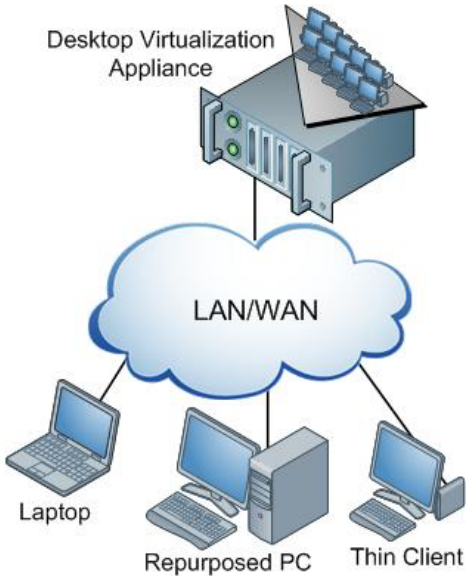

Figure 3.2: Desktop Virtualization

through a service provider that provisions desktop as a service (DaaS). Host-based virtual machines allow users to connect to a single VM hosted on a server which is either a persistent desktop or a random VM from an available set of VMs running the same OS. The shared hosted model allows users to connect to a single shared desktop or individual applications running on the same desktop.

Client-based Virtual Desktops

Client-based processing uses the local hardware to run part of the OS while the disk image is synced with a remote server. One type of client-based desktop virtualization is called OS streaming where the OS runs on local hardware but boots from a remote disk image connected via network. In this use model, multiple users may boot using the same disk image, but the local hardware needs to be a full hardware client in order to run the OS.

Another type of client-based virtualization is using a virtual machine on a full hardware client where the virtual machines are managed by syncing the OS disk image with a central data server. Here, a network connection is only needed when the disk image syncs to the server, so the virtual machine on the client is able to run offline if needed.

Advantages and Disadvantages

Many of the use cases of server virtualization also play a part in desktop virtualization, including remote access to data and applications as well as improved consolidation of resources and data. The main difference is the presented information in desktop virtualization, specifically the personalization of a workspace along with the applications and files that a user may have on their particular desktop.

Desktop virtualization is an important aspect of server virtualization because of the beneficial integration with the modern computer and its user. Most users now have their own local computers in the form of workstations or laptops with a network connection. By hosting desktops on centralized servers, the server can provide a service to each user's device with the needed computational performance and resources while still being secure. IT need only provide support and maintenance on the servers to distribute necessary patches and updates to all clients connected to their system, and users can still use their devices and access their files without having to update hardware. If their client lacks the performance needed for some intensive applications, those workloads can be offset by server resources. Also, because desktops are virtualized, legacy and incompatible applications can also be provided to clients without the need to port those applications to new hardware or operating systems.

Provisioning available hardware over a network connection also enables workers to securely work on-site or remotely with the tools and data that they need. Even contractors who may only be on-site for a period of time may be given a virtualized desktop to use so that they can securely work without the risk or cost-concerns. Normally IT personnel would need to maintain and secure a workstation or loaner laptop for the duration of the contractor's work period, but with desktop virtualization, contractors and partners are able to access the applications and desktops they need without compromising security of the business's network. Finally, workers are able to bring in their own devices that they can use as thin clients to access a virtualized desktop securely without the cost of dedicated network infrastructure or risk usually associated with non-corporate devices that may be exposed to security risks outside of the workplace.

The limitations of desktop virtualization are multi-faceted and result from improper setup of the virtual system and compatibility with client devices. Security may be a major issue, where loss of privacy and autonomy are issues if the hosting server and storage is not properly setup. This may happen when the host system does not differentiate and partition users to their own file system. If the VM is not persistent across uses, users will not be able to customize their desktop and will have to use the predetermined setup, however persistent configuration files fix this problem on most desktop virtualization servers. While running a desktop OS on your hardware allows the user to utilize hardware such as printers, device drivers may show a challenge for virtual desktops that simply deliver a window for the user to interact with. Users will have to contact their system administrator to configure devices to work with the system and this process may not resolve if the drivers are not compatible with the specific OS instances running on the virtualization server. Networking is the most limiting resource when running an desktop virtualization server because in order for the end user to have a good experience, low latency between the client device and server must be ensured. Otherwise, the user will experience input and response latency that will make the system seem like it is not working properly.

Operating System-Level Virtualization

While OS-level virtualization may seem very similar to desktop virtualization semantically, OS-level virtualization works a bit differently. Instead of having a host/guest model as with full virtualization, OS-level virtualization uses a single host OS that provides OS-level functionality to each of its users. In short, a single OS provides the infrastructure for all of the virtualization. Each user is partitioned into its own isolated container or zones so security breaches or resource deadlocks don't affect other users. Using OS-level virtualization, users are able to share the common binaries and libraries of the host OS so that many users are able to multiple programs at the same time without interfering with one another.

Solaris zones utilize this virtualization model to provide servers with a virtualized operating system environment with the use of one operating system. Figure 3.3 shows the architectural model of OS-level virtualization where each guest has its own file system. IP

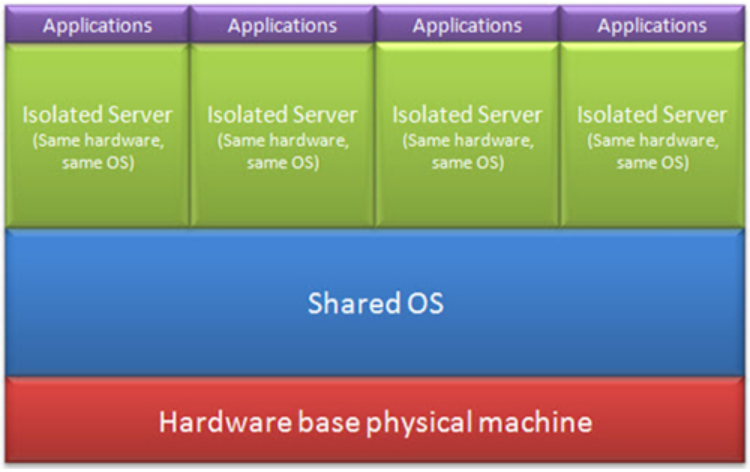

Figure 3.3: OS-Level Virtualization

address, server configuration file, and its own space to run completely different applications.

Advantages and Disadvantages

Using only the host OS to manage the needed guest instances, OS-level virtualization is lightweight and fast and can support a large number or guest instances. The main reason why OS-level virtualization is fast, as compared to full virtualization, there is no hypervisor level the guest systems need to go through to process hardware requests. However, because there is a single host OS that handles all of the virtualization, each guest system or instance and its applications must support the host OS, else they will not run on this model. Other issues include making sure that each guest is isolated from one another and securing each user's data in a way so that only the user is able to access their own data.

Chapter 4

Resource Virtualization Applications

Storage Virtualization

The Storage Networking Industry Association (SNIA) defines storage virtualization as the abstraction or hiding of a storage system or service from applications, host computers, or general network resources, for the purpose of enabling application and network-independent management of storage or data. Thus much like hardware virtualization, the purpose of storage virtualization is to hide the complexity of the underlying or lower level resources to add capabilities to this system by aggregating the functions and devices involved in using and maintaining a storage system. Storage virtualization occurs both locally on end user computers and on remote servers.

Local Storage Virtualization

For end users, disk drive virtualization, file systems, and RAID (redundant array of independent disks) systems are the most prevalent use of storage virtualization on local computers or local area networks (LANs).

The simplest form of storage virtualization is disk drive virtualization, where at the lowest level, a "disk is defined in terms of cylinders, heads, and sectors." These physical parts of the disk are virtualized by firmware from addresses into logical blocks that are use-able by the OS and applications. Even if defects in the disk occur where the disk can no longer store data, the firmware re-maps those blocks to other free blocks on the disk. This remapping is done so that the end user does not have to manually ensure that their

data is uncorrupted, nor keep track of all the "bad" blocks on the disk.

Looking at a file system, there is a hierarchical structure of directories and files that correspond to data stored on a hardware storage drive. The data on the drive, which consists of bits stored on magnetic or flash memory, is put into logical blocks that the OS can read based on a specific file structure format such as New Technology File System (NTFS), or inodes. Although the data may seem contiguous when looking at the file system on the terminal or graphical user interface (GUI), the physical bits may be spread out throughout the disk on separate logical blocks. The disk firmware handles the reading and writing to these blocks while the OS processes and presents the data in a readable format for the end user.

Another form of file system virtualization is the concept of network attached storage (NAS). This form of virtualization uses file servers that manage shared access to the storage over a network connection. Wherever the physical location of the file is in that storage, different computers with possibly different operating systems are able to access the data through the same file system. The abstraction of data access to the networked storage give the files "location transparency"; the files appear as if they were local files.

In some end user computers, RAID systems are implemented to combine drive components into logical drives for the purposes of data redundancy or increased disk performance. For example, a RAID 0 array combines two or more physical drives into one logical drive that is the sum of all capacities of the disks in the array. This setup is purposed to increase read and write speed by using all disks at the same time to process data at the expense of data recoverability.

With RAID systems, file systems, and disk drive virtualization, users are able to logically organize data so that it is easier to handle by abstracting the lower level details that are not always needed.

Remote Storage Virtualization

Remote storage or rather, cloud storage, is an application of storage virtualization that physically lies within data center servers accessible over the Internet or a network connection. These remote servers utilize all levels of storage virtualization including drive virtualization,

file systems, and block virtualization but on a much larger scale so that storage needs for multiple businesses and users can be met.

File virtualization is commonly deployed in a data center where data may be stored over long periods of time with the need of fast retrieval and availability. Automated migration transfers rarely used data to inexpensive secondary storage media such as optical disks or high density disk storage like hard disk drive arrays. Users and applications are able to access the data as if it were still on a primary storage medium. A pointer in the file system combined with metadata allows the data to be located without the user knowing where the exact location of the file is. This is called a Hierarchical Storage Management (HSM).

Generally, when people are talking about storage virtualization in the data center, they are referencing block virtualization. Block virtualization is the virtualization of many physical storage drives, both magnetic disk and flash drives, into logical drives that require no more additional intelligence to access that drive. Simply put, the drive that the users or application sees is a larger disk and this is called block aggregation. By adding RAID systems or clusters with parity and/or striping, these blocks can not only grow in size by adding more drives but also improve visible performance and availability to the end user. Here the physical details of the storage system are not needed, the user and application simply need a drive of a certain size, speed, and reliability with the possibility of scalability.

Storage Area Networks

Storage area networks (SANs) are server level appliances that provide storage infrastructure to a number of servers over a high-speed (sub)network. The appliances may be anything from a commercial platform or a proprietary hardware design that use some level of virtualization that allow for management, redundancy, and failover. These SANs may be integrated into existing data centers using symmetric (in-band) and asymmetric (out-of-band).

Symmetric virtualization of SAN appliances place those SAN devices between the server and the storage array. The servers treat the SAN appliances as I/O targets to make data requests, which shows only the resources that are assigned to each host. The symmetric approach presents the SAN as a standard I/O device to the host, so no special drivers or software is needed by the host to use this method. The plug-and-play aspect of symmetric

appliances also means that a number of different host operating systems are supported.

Out-of-band (asymmetric) virtualization, places the SAN device outside of the data path between the storage array and the hosts. The hosts contact the SAN device to access the storage array. This storage model needs the hosts to be aware of the virtualized storage via a virtualization client or special drivers on the host. This client on the host receives the information from the SAN appliance about the logical drives and formatting of those drives, which then allows the host to access their storage array. The client is also responsible for security, snapshots, and replication because the SAN appliance in this method is unable to monitor the I/O of the host.

Network Virtualization

In the area of networks, there are many applications of virtualization that involve the obfuscation or abstraction of lower level detail from the higher level applications or users. In fact, the OSI layer model of networking uses that theory of abstraction so that the lower layers take care of the lower level details such as the mapping of bits to the physical layer and detailing the network through data links and nodes. The abstraction of detail at the upper layers allows for a more robust and concise operation while relying on the layers underneath to provide the necessary interfacing for the information to enter the network.

Abstraction can apply to the details of the resources or network hardware that is given a logical form instead of relying on the physical interface to present data to the upper layers. By translating the physical resources into logical resources, we can make those resources more robust by providing redundancy, failover, and flexibility.

The virtual circuit concept forms the basis of our modern packet switching network that treats a network connection as a uni/bi-directional channel between two devices. This concept allows networks to put many pairs of devices on the same virtual connection that may actually be a number of different connections. The opposite may also be true, where a single device is able to access many other devices over a single network connection but is really making many logical connections to many other devices.

Virtual Private Networks

Virtual private networks (VPNs) use tunneling protocols to allow remote user access to a network. In this method, the user appears to be connected locally to the network when they may be in a distant location. These tunneling methods include Secure Internet Protocol (IPSec), Point-to-Point Tunneling Protocol (PPTP), and Layer 2 Tunneling Protocol (L2TP).

The topic of virtual private networks include a variety of topics that describe how modern networks apply virtualization to transform physical connections into logical connections. These topics include Virtual LANs, Provider Backbone Transport (PBT), Simple Traversal of UDP through NATs (STUN), link aggregation, and Virtual Private LAN services (VPLS).

Virtual LANs

Virtual LANs also use abstraction to create and manage logical connections between devices on a network. Network administrators are able to distribute VLAN IDs based on user groups and this allows for users to access data on their VLAN and devices as if they were connected through the same switch. The VLAN concept means that devices are no longer constrained to their physical location on the network, instead they are logically determined by a network administrator. This allows for more flexibility in configuring the network so that users can be separated into distinct broadcast domains where they can be constrained to certain parts of the network without interfering in other VLANs. For example, if a remote salesperson needs to log into the enterprise network to check product pricing, then their VLAN ID will allow them to connect to the network and check the product listings but will not allow them to access the HR VLAN to see those departmental emails.

At the public network level, VLAN IDs are used to separate customers' networks from each other on the public network. Here, Internet Service Providers (ISPs) are able to separate traffic intended for different customers at layer 2 based on VLAN ID and MAC addressing.

Provider Backbone Transport (MAC-in-MAC)

Provider Backbone Transport (PBT) is defined in IEEE 802.1ah as a way to encapsulate Ethernet frames with another MAC header. This separates the customer's networks from the providers' networks, so the service providers can provision paths over the backbone using their own devices' MAC addresses. This practice eliminates the learning the location of every single MAC address that is connected to the network, instead network administrators can configure forwarding tables to create a path for a range of VLAN IDs and MAC addresses.

STUN and NATs

In STUN, network addresses and ports are mapped to virtual address and ports internal to a gateway. The Network Address Translator (NAT) will provide the mapping of external addresses and ports to internal ones so that devices external to the network can communicate with devices behind the NAT. The NAT provides protection to devices on its network using firewall so that packets are checked before entry to the network. The NAT also prevents malicious attacks by external devices by presenting the external host with a virtual address and port that allow connection to the internal STUN server but does not allow for direct connection to the server itself. This prevents a user from accessing private files while still being able to connect and use services provided by the server.

Link Aggregation

IEEE standard 802.3ad defines the link aggregation standards. Link aggregation is implemented by connecting two or more physical ports in parallel to provide redundancy and better throughput for one logical link. Split Multi-Link Trunking (SMLT) and Distributed Split Multi-Link Trunking (DSMLT) allow for these ports to be spread across multiple switches to increase the number of ports available for redundancy or throughput. Virtualizing links this way allows for more flexibility in speed and failover protection, so that links may increase their speed when needed or provide a method of backup if one of the ports go down.

Virtual Private LAN Services

Virtual Private LAN services (VPLS) provide LAN services to multiple networks over connections called pseudo-wires. These pseudo-wires provide the means for multiple networks to create a bi-directional channel of communication to setup a mesh network needed to simulate broadcast domain over multiple autonomous networks. In the provider network, paths simulate bridges or switches using IP and MPLS to direct traffic as if the packets were on a LAN using VPN tunneling. Another benefit of using pseudo-wires is that if those connections go down in the provider network, backup paths already exist to service the packets. The simulation of a LAN network simplifies the network connections over the provider network and provisions a means for customers to treat different Ethernet networks as one network. VPLS are a much more reliable solution than simple WAN links that without the overhead of establishing a VPN for each customer network connected in the mesh.

Chapter 5

Conclusion

Future Aspects of Virtualization

The future of virtualization is happening right now with companies like Citrix, VMware, Microsoft, and other cloud-based computing platforms like Amazon's AWS and Google services. These companies are at the forefront of providing enterprise-level virtualization solutions to businesses and users across the world. Using virtualization technologies businesses are able to solve IT problems, like server consolidation and hardware/software refreshes, while reducing business expenses and complexities, such as downtime and energy costs. Users can also adapt their computing environment to take their work and data with them wherever they go.

Other advances in virtualization are more specific, using specialized hardware, such as NVIDIA's GRID technology, to provide powerful 3D applications to power users and designers. As the world becomes more and more networked, the need for computing devices for new users grows. However, no longer are people restricted to large and powerful computers to achieve their goals – virtualization provides even smartphones and tablets with the processing power of a server-based architecture.

As a whole, virtualization has become tightly integrated with much of the modern computing environment from hardware to networking. Its purpose is to provide an abstraction of detail from the underlying hardware and physical devices so that they are more robust in their operation and maintenance. Everyday, the gamut of personal computers to mainframe systems use virtualization to provide the resources for their applications and operating systems. Virtualization also provides the necessary resources for users to control

their computing experience and not let their applications or data control how they use their devices. In the future, we will need virtualization to transcend the physical limits of hardware to provide a scalable computing experience to all users without expanding beyond the physical limitations of space and energy.

Chapter 6

Appendix: Abbreviations

API: Application programming interface
BIOS: Basic input/output system
BSD: Berkeley Software Distribution
CPU: Central processing unit
DaaS: Desktop-as-a-Service
DSMLT: Distributed Split Multi-Link Trunking
GUI: Graphical user interface
HSM: Hierarchical Storage Management
I/O: Input/output
IEEE: Institute of Electrical and Electronics Engineers
IP: Internet protocol
IPSec: Secure Internet Protocol
IPC: Interprocess communication
IT: Information technology
L2TP: Layer 2 Tunneling Protocol
LAN: Local area network
MAC: Media Access Control
MPLS: Multi-Protocol Label Switching
NAS: Network attached storage
NAT: Network address translation
NTFS: New Technology File System
OS: Operating system
OSI: Operating Systems Interconnection
PBT: Provider Backbone Transport
PPTP: Point-to-Point Tunneling Protocol
PV: Paravirtualization
RAID: Redundant array of independent disks
SAN: Storage area network
SMLT: Split Multi-Link Trunking
SNIA: Storage Networking Industry Association
STUN: Simple Traversal of UDP using NATs
VDI: Virtual desktop infrastructure
VLAN: Virtual local area network
VM: Virtual machine

VMM: Virtual machine monitor
VPLS: Virtual private LAN services
VPN: Virtual private network
WAN: Wide area network

Bibliography

[1] Altaf Al-Amin. Introducing microsoft application virtualization (app-v), April 2009.

[2] Z. Amsden, D. Arai, D. Hecht, A. Holler, and P. Subrahmanyam. Vmi: An interface for paravirtualization. *Proceedings of the Linux Symposium*, 2:371–386, July 2006.

[3] Howard Baldwin. Network virtualization vs. software-defined networks: What the heck is the difference?, October 2014.

[4] Darryl Chantry. Mapping applications to the cloud, January 2009.

[5] Citrix. Session virtualization and app virtualization with xenapp.

[6] Virtual apps and desktops delivered by citrix meet today's business demand for anywhere access.

[7] Abhishek Ghosh. What is hardware-assisted virtualization, September 2014.

[8] K. Scarfone; M. Souppaya; P. Hoffman. Guide to security for full virtualization technologies- recommendations of the national institute of standards and technology. *NIST Special Publication 800-125*, January 2011.

[9] Bill Kleyman. Hypervisor 101: Understanding the virtualization market, August 2012.

[10] Eric Knorr. What desktop virtualization really means, May 2010.

[11] Microsoft. Applications virtualization.

[12] Alan Murphy. Virtualization defined- eight different ways. Technical report, F5 Networks, https://www.f5.com/pdf/white-papers/virtualization-defined-wp.pdf.

[13] F. Bunn; N. Simpson; R. Peglar; G. Nagle. Storage virtualization. Technical report, SNIA, http://www.snia.org/sites/default/files/sniavirt.pdf, 2003.

[14] Paravirtualization (pv).

[15] Domagoj Pernar. Virtual applications (app-v) download repository, October 2009.

[16] Mehdi Rahimi, S.; Zargham. Security implications of different virtualization approaches for secure cyber architectures. *Secure and Resilient Cyber Architectures Conference*, October 2010.

[17] Margaret Rouse. server virtualization, June 2009.

[18] Margaret Rouse. app virtualization, November 2011.

[19] Margaret Rouse. desktop virtualization, November 2011.

[20] Tim Sommer. Top 200 sam terms- a glossary of software asset management terms. Technical report, OMTCO, June 2012.

[21] M. Taillefer, B. Silva, S.W. Adermann, and L.M. Dyer. Partial virtualization on computing device, May 25 2010. US Patent 7,725,305.

[22] Virtual machines.

[23] VMware. Understanding full virtualization, paravirtualization, and hardware assist. Technical report, VMware, http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf, 2007.

[24] What is desktop virtualization?

[25] Candid Wueest. Does malware still detect virtual machines?, August 2014.