Full Virtualization Hypervisors

Justin Kerchal **Adam Keaveney** **Jim Xu**

ABSTRACT

The basic principle of a hypervisor is that there is a piece of software that interacts directly with the hardware. This piece of software allows for multiple virtual machines to be running on the same hardware at the same time. The hypervisor is the 'host' and the VMs are the "guests'. Each VM thinks that it is the only machine there and thinks it is talking directly to the hardware. In reality it is actually talking to the hypervisor which has allocated the resources that the VM is allowed to use. As technology advance, hypervisors are also built on host operating system to give users a better access for multiple operating systems. This type of hypervisors do not have directly physical hardware access but have overall better software performance.

Keywords

Full virtualization, Hypervisors, type-1 hypervisors, type-2 hypervisors, Xen HVM, VMware, KVM.

1.INTRODUCTION

In order to explain how a hypervisor can be used a basic understanding of how one works must first be obtained. A modern desktop computer or laptop has the basic components including a motherboard, CPU, some RAM, a video card, and a power supply. These pieces of hardware are capable of doing more work than one operating system requires. The CPU in particular spends most of its time waiting for instructions. A hypervisor is a piece of software that interacts directly with the hardware of a computer and allows a single computer to run multiple operating systems simultaneously [1]. The physical hardware is called the host and the virtual machines (VMs) running on the host are called guests or guest operating system (Figure 1). Thanks to the way in which a hypervisor distributes resources and interacts with the VM, a VM thinks that it is the only machines there and that it has exclusive access to the resources it can see. Every VM running on the host is presented with this same illusion. In reality, the hypervisor software is managing resources for each guest and the CPU is spending more time executing instructions instead of waiting. This makes a hypervisor much more efficient than a traditional computer.

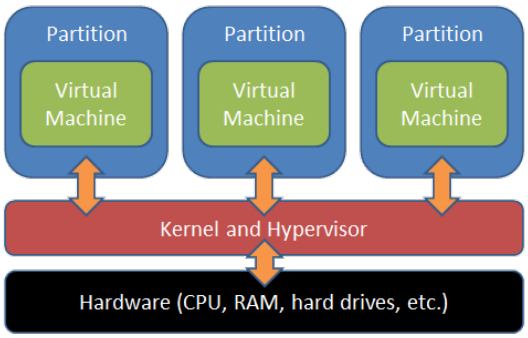

Figure 1. basic idea of hypervisor

This concept of having multiple computers 'inside' a main computer is especially useful when it comes to larger amounts of users. In the world of today, there are more devices than ever before. Most every person has multiple computers, each of them running an operating system. With all of these machines, space has become a commodity. If an office of 100 people requires a computer for each worker, that is 100 machines that need to be maintained and supported. Each machine might have different specifications and requirements and when maintenance is factored in, this is a lot of work for a sysadmin or IT department to do. If all the of machines were instead built in a hypervisor, then each worker only needs a small thin-client with an internet connection to do their work because it's all done on virtual machines. A thin client is a basic small form computer with enough hardware specifications to display images on the screen and access the internet. All the actual computing is done in the virtual machine that the thin client is used to log into. Since all the VMs are hosted on the same hardware all the VMs will share the same hardware, so hardware inconsistencies between machines are non-existent. To fix an issue for a user, the sysadmin simply logs into the host physical machine and does what needs to be done. There is no more running from desk to desk to help users. This document will explain why virtualization is needed today, what it can do tomorrow, and why it is worth pursuing in the future.

2.APPROACHES TO PROBLEM

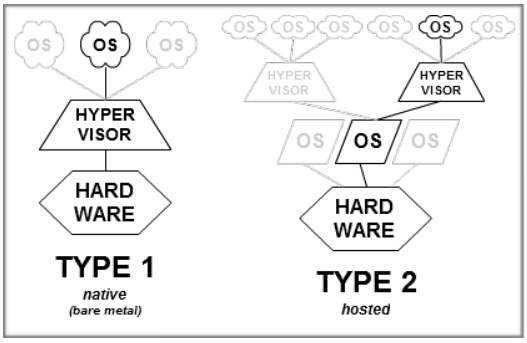

Figure 2. two types of hypervisors

Gerald and Robert, in their article, *Formal Requirements for Virtualizable Third Generation Architectures*, classified two types of hypervisors.(Figure 2)

Type-1 is native or bare-metal hypervisor, it runs directly on host hardware to control the underlying hardware as well as manage guest OS. This is a technique where the abstraction layer sits directly on the hardware and all the other blocks reside on top of it. The Type 1 hypervisor runs directly on the hardware of the host system in ring 0. The task of this hypervisor is to handle resource and memory allocation for the virtual machines in addition to providing interfaces for higher level administration and monitoring tools.The operating systems run on another level above the hypervisor. Modern equivalents include Xen, Oracle and Microsoft Hyper-V. Another one is hosted hypervisor (type-2 hypervisor), it runs as a process on the host. Guest OS Virtualization is perhaps the easiest concept to understand. In this scenario the physical host computer system runs a standard unmodified operating system such as Windows, Linux, Unix or MacOS X, and the virtualization layer runs on top of that OS being in effect a hosted application. In this architecture, the VMM provides each virtual machine with all the services of the physical system, including a virtual BIOS, virtual devices and virtual memory. This has the effect of making the guest system think it is running directly on the system hardware, rather than in a virtual machine within an application.VMware is an example of type-2 hypervisors. However, Kernel-based Virtual Machine (KVM) can act as type-1 and type-2 hypervisors at the same time [2].

2.1 Type-1 Hypervisor (native or bare-metal hypervisor)

Xen is a virtual open source project developed by the Computer Laboratory of the University of Cambridge. Xen can safely execute multiple virtual machines on a set of physical hardware, which is extremely close to the operating platform and has the least resources. Xen is known for its high-performance and acquiring less resources, and thus wins a high degree of recognition and support from IBM, AMD, HP, Red Hat and Novell and many other world-class hardware and software manufacturers. Xen, based on X86 architecture, is a virtualization technology that has fastest growing speed, the most stable performance and minimal sources. Novell SUSE Linux Enterprise Server is the first to use Xen virtualization technology. It is particularly suitable for server application integration, which can effectively save operating costs, improve equipment utilization, maximize the use of data center IT infrastructure.

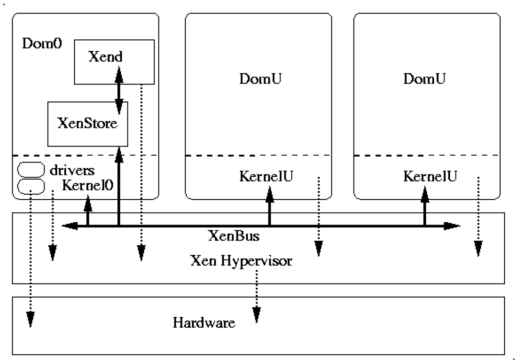

Figure 3. the Xen(Type-1) architecture

In the Xen environment, there are two main components (Figure 3). One is the virtual machine monitor (VMM), also known as hypervisor. The hypervisor layer is the first layer of hardware that must be loaded first between the hardware and the virtual machine. Once the hypervisor is loaded, you can deploy the virtual machine. In Xen, the virtual machine is called "domain". In these virtual machines, one of them plays a very important role, is domain0, has a very high privilege. Typically, the operating system installed before any virtual machine has this privilege [3].

Domain0 is responsible for some specialized work. Since the hypervisor does not contain any drivers for hardware conversations and interfaces to the administrator, these drivers are provided by domain0. With domain0, administrators can use some Xen tools to create other virtual machines (the Xen term is called domainU). These domainU are also called unprivileged domains. This is because in the x86-based CPU architecture, they will never enjoy the highest priority, while only domain0 can [3].

In domain0, an Xen process is also loaded. This process manages all other virtual machines and provides access to these virtual machine consoles. When creating a virtual machine, the

administrator uses the configuration program to talk directly to domain0 [4].

As the Xen approach to virtualization has taken a big step forward, the founders of Xen set up their own company, XenSource, which has been acquired by Citrix later. The goal of XenSource was to provide a complete virtualization solution based on the Xen hypervisor in order to compete with other virtualization products such as VMware. Other companies also have integrated Xen hypervisors in their own products. For example, Linux manufacturers Red Hat and Novell have included their own versions of Xen in their operating systems. Since most of Xen is open source, these solutions for virtualization are very similar.

2.2 Type-2 Hypervisor (hosted hypervisor)

VMware (Virtual Machine ware) is a "virtual PC" software company, providing virtualization solutions. VMware virtualization directly introduces a streamlined software layer into the computer hardware or above host operating system. It contains a dynamic and transparent way to allocate hardware resources of the virtual machine monitor. In order to achieve multiple operating systems running on the same physical machine at the same time, sharing hardware resources with each other is applied.

VMware first introduced virtualization technology on x86 computing platform in 1999. VMware virtualization pulled the operating system away from the underlying hardware on which it was running and provided standardized virtual hardware for the operating system and its applications. As a result, a virtual machine can run one or more shared processors at the same time independently.

VMWare products enable to run two or more Windows, DOS, and LINUX systems simultaneously on a single machine. Compared with the “multi-boot” system, VMWare uses a completely different concept. Multi-boot system can only run a system at a time, the system switches need to restart the machine. VMWare is truly “running” at the same time, multiple operating systems on the main system platform, switching to Windows as standard application. And in each operating system you can carry out a virtual partition [5]. The configuration does not affect the real hard drive data, so you can use network interface card to connect several virtual machine into a local area network, which is extremely convenient.

The software layer contains a virtual machine monitor (or “hypervisor”) that allocates hardware resources dynamically and transparently (Figure 4). Multiple operating systems can run

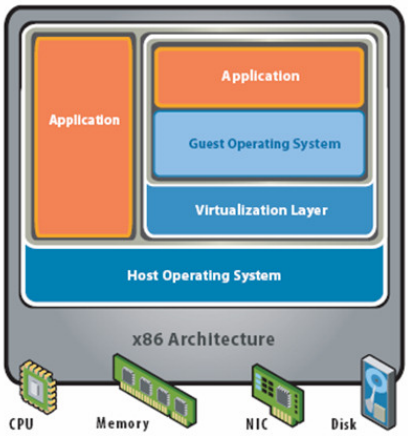

Figure 4. the VMware(Type-2)

concurrently on a single physical machine, sharing hardware resources with each other. Virtual machines are fully compatible with all standard x86 operating systems, applications, and device drivers because the entire computer, including the CPU, memory, operating system, and network devices, is packaged. You can safely run multiple operating systems and applications on a single computer at the same time, and each operating system and application can access the resources it needs when needed [6].

In all virtualization software technology to optimize and manage the IT environment, VMware virtualization technology has been the most widely used, from the desktop environment to the data center are involved.

2.3 Type-3 Hypervisors (A combination of type-1 and type-2)

KVM is a hardware-based virtual machine (VM) implementation proposed by an open-source organization called Qumrant in Israel in October 2006. The Linux 2.6.20 kernel, released in February 2007, is the first time that included KVM. In fact KVM is only part of the virtualization solution, and it needs the underlying processor support for multiple operating systems to provide virtualization processors.

KVM is an open-source Linux native full-virtualization solution based on virtualization extensions (Intel VT or AMD-V) for X86 hardware. In KVM, virtual machines are implemented as regular Linux processes that are scheduled by standard Linux dispatchers; each virtual CPU of the virtual machine is implemented as a regular Linux process. This allows KVM to use the existing functionality of the Linux kernel [7]. As the virtualization hardware to provide a new framework to support the operating system directly in the above operation, without the need

for binary conversion, reducing the associated performance overhead, greatly simplifying the VMM design, making VMM performance more powerful. Beginning in 2005, Intel promotes Intel Virtualization Technology in its processor product line, which is called IntelVT technology.

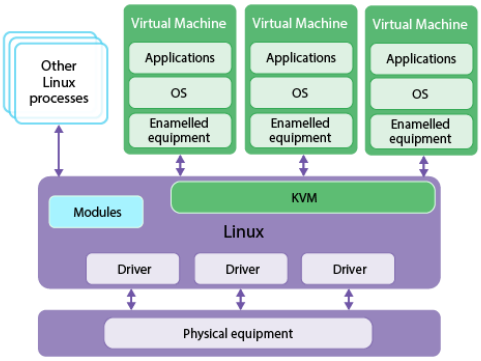

Figure 5. the KVM architecture

KVM is a software that allows to implement computer-based virtualization in OS Linux and Linux-like systems (Figure 4). For some time now KVM is a part of Linux-kernel, that is why they develop together. It works only in systems with hardware virtualization support on the CPU Intel and AMD. For organization of work KVM uses direct access to a kernel with CPU-specific module (kym-intel or kvm-amd). Moreover, the complex contains a main kernel kvm.ko and elements UI, including popular QEMU. Hypervisor enables to work directly with virtual machines files and disc images from other programs. Isolated space is created for every machine with its own RAM, disk, network access, video card and other devices [8].

In addition to CPU virtualization, memory virtualization is also achieved by the KVM. In fact, memory virtualization is often used by virtual machine to achieve the most complex part. CPU memory management unit MMU is through the form of the page table to run the virtual address into the actual physical address. In virtual machine mode, the MMU's page table must perform two address translations in one query. Because in addition to converting the virtual address of the client program to physical address of the client, it also needs to convert the client physical address into a real physical address. Since KVM is only a simple virtualization module, its memory management is not self-realization, and needs the help of Linux kernel memory management. KVM can use any storage supported by Linux, in the realization of the driver, directly with the help of the Linux kernel to drive any hardware. KVM architecture is very simple, is a module of the kernel. The user provides that to the virtual machine to use through QEMU simulation hardware. A virtual machine is a common Linux process, through the management of this process, you can complete the management of the virtual machine.

2.4 Contrast between different types of hypervisors

Table 1. differences between hypervisors

Type 1 Advantages	Type 1 Disadvantages
Increased security	Need particular HW component
Allows higher density hardware	Strict HW requirements
Hypervisor has direct access to HW	Costly

Type 2 Advantages	Type 2 Disadvantages
Host OS controls HW access	Decreased Security
Ease of Access	Lower VM Density
Allows for multiple OS's	Needs a host OS first

Generally, type-1 hypervisors provide better performance and flexibility than type-2 hypervisors, because it runs directly on hardware and exposes hardware resources to virtual machines (VMs), thus reducing the overhead that caused by the executing hypervisors. Since it is completely independent from the Operating System, one virtual machine or guest operating system do not affect the other guest operating systems running on the hypervisor. The hypervisor is small as its main task is sharing and managing hardware resources between different operating systems, and it allows higher density hardware. However, type-1 hypervisors need particular hardware component to support different operating system, which causes it is hard for manufactured. The strict hardware requirements also make it costly compared to type-2 hypervisors [9].

Type-2 hypervisor is also known as Hosted Hypervisor, because it runs on the original operating system as a process. In this case, the hypervisor is installed on an operating system and then supports other operating systems above it, and that makes it completely dependent on host Operating System for its operations. The host operating system have the privileged control over the hardware access, and guest operating system can be only have a few hardware support. This makes type-2 hypervisors have easy access of different operating systems without changing the host system through restarting machine. On the other hand, while having a base operating system allows better specification of policies, any problems in the base operating system affects the entire system as well even if the hypervisor running above the base OS is secure [9]. That means type-2 hypervisors need a powerful and secure host operating system first.

Typically, a Type 1 hypervisor is more efficient than a Type 2 hypervisor, yet in many ways they both provide the same type of functionality because they both run the same kind of VMs. In fact, you can usually move a VM from a host server running a Type 1 hypervisor to one running a Type 2 hypervisor and vice versa. A conversion may be required, but the process works. Type 1 hypervisors support hardware virtualization. Because they run as an application on top of an operating system, Type 2 hypervisors perform software virtualization [9].

3.LIMITATIONS

While virtualization is a fantastic tool that can be widely employed, it is not suitable for every task. According to Scott Matteson, a senior systems engineer and contributor to the online Systems Admin blog Tech Republic, there are some things that virtualization is not recommended for:

- Anything with a dongle/required physical hardware
- Systems that require extreme performance
- Anything on which your virtual environment depends
- Anything that must be secured

While support for serial devices attached to virtual machines has improved in recent years, it is still difficult to get the dongle to 'play nice' with the VM. A technique called 'serial pass-through' has to be used and it does not always function or can be disabled after an update or a reboot of the physical host machine.

If something needs to be fast, it is put closest to the CPU. A normal native OS talks directly to the hardware and is therefore relatively quick. If a process is very I/O demanding, the native OS can easily talk directly to the disk and read/write as needed. A VM is separated from the hardware by the virtualization software layer. While still operating with near imperceptible speed differences, the guest OS still takes more time to do things like I/O. If a system needs to be as fast as possible, virtualizing it is not the best course of action.

Any sort of 'dependency loop' is going to cause problems eventually. Matterson uses the example of having the virtual domain controller (DC) being required to log into the virtual management environment [10]. If the virtual DC is down, it is not possible to log in and fix the DC because the management environment requires the DC to be up.

While a VM itself is quite secure, at the end of the day it is still just a file on a disk. A malicious user could simply copy the VM file, modify it to suit their nefarious purposes, and replace the actual VM with the malign one. It is quite difficult to physically secure a virtual object, but with improved physical security and limited access the host machines this can be alleviated.

In the future, the time it takes for a VM to communicate with the physical hardware will become less and less. Faster hypervisor technology will ultimately make VMs fast enough that there will be little to no differences in speed and then they can then be used for heavy I/O applications as well. Improved physical port passthrough in the future will allow for things like security dongles and authenticators to be used on a virtual machine instead of an often aging and dying physical machine.

Making virtualization software that is less focused on industry and geared more towards the everyday consumer may also start to appear on the market. Often a consumer has one type of system, and would like to run software X but the OS they have does not support software X. They could instead turn on a virtual machine and use software X inside the VM all without having to get a new physical piece of hardware. People may someday have a single server in their house that is running some sort of virtualization technology on it and each room in the house will just have a terminal so they can log into the appropriate VM.

4. The PRESENT AND FUTURE

Hypervisors have indeed come a very long way. The technology has revolutionized the Computer Science industry, centralized security presents the opportunity for small and large companies to outsource security to an externally hosted hypervisor. It is empirically supported that many the Computer Science industry is moving towards both cloud and non-cloud supported fully-virtualized workspaces. The amount of money saved on security alone is worth the migration [11][12].

The start of the art is constantly changing, in the form of added features and flexibility, however, the technology will exceed its own potential [13]. We will eventually see the costs of server maintenance overcome the cost of software production. Competition is high in this market and while it is likely the reason hypervisors have come so far in the past decade, it will also be the reason for the decline in the market (in the coming years). Susan Gartner says "Despite the overall market increase, new software licenses have declined for the first time since this market became mainstream more than a decade ago [14]. Growth is now being driven by maintenance revenue, which indicates a rapidly maturing software market segment". This means that intuition coming out of young and experienced software engineers has less of a grasp on the industrial hypervisor machine than a cloud based

server that requires maintenance on a weekly, if not daily, basis. The future is not grim though, it just paints a strangely unnerving picture of a tech-world governed by huge corporations able to bully out competition through capital banking. For now, though, it seems that competition will beat the prices. The market is unpredictable (except that overall the money is going to virtualization and cloud computing), VMware, the leader of the virtualized world, is constantly striving to hold the reigns. "VMware has the largest market share when it comes to virtualization – 60%, per research firm IDC. VMware global field CTO Paul Strong says that today most adoption is being driven by consolidation and the desire for private and hybrid cloud environments. But market share does not mean VMware can rest on its laurels. New demands, such as the software-defined data center (SDDC), are pushing virtualization technology forward". It's clear that, to stay ahead of the game, the prices need to be down, and software innovation needs to be in tip top shape. This further supports the idea that the future will be decided by whoever holds the largest most efficient database centers i.e. who has the deepest pockets [15].

However, there are a few other factors to consider when looking at the present and future of full-virtualization. The idea behind such a thing is not basic, however, it is entirely feasible that a small company could design and implement their own hypervisor. Today we see all the largest players in the game moving towards grabbing a stake in the hypervisor world. Oracle, Citrix, Microsoft, Amazon, the list goes on and on. Many independent companies have emerged and made their name through this industry (Exhibit A, VMware). The truth is, developing and refining a hypervisor is complicated and costly. This leaves little room for the flexibility one would need when providing a product (hypervisor) to a wide range of customers. Network world published: "This is an exciting moment for data virtualization. The options available for virtualization are expanding, and are providing advances in processing speed around big data and data integration. This is just one of many areas around virtualization getting attention…and usually with the words "new" and "future" close by". This is simply the perfect statement to illustrate the point that the future for the software layer between user and hardware will diversify the competition in this industry. A multi-billion-dollar company can buy up all the server power they want, yet, it's impossible to assume the needs of every customer. The future poses a constant wrestling match between hypervisor and cloud computing companies. All this revolves around the struggle to create flexibility amongst the different types of processing that customers need to do daily. CPU utilization requires different algorithms (night and day) to efficiently process data queries from, for example, a large warehouse management company vs. a fully virtualized online gaming provider. The requirements of the two examples could never be managed by the same hypervisor. This provides an awesomely bright light at the end of the tunnel... a future of equal competition and opportunity perhaps.

In the next few years it's easy to suggest where the market will go and who might control it etc. Judging what the state of the art will look like poses a more difficult task. The fact of the matter is that virtualization has come and gone and is now making a decade long comeback in a way we haven't seen before. Hypervisors are in for the long haul. Today we see improvements in Consolidation and Containment, Infrastructure provisioning, Business simplification, the list goes on. Tomorrow suggests any number of things, it's not a stretch to see a hypervisor-like technique applied to mobile phones. With the huge number of transactions mobile phones make to cloud servers daily, why not get each memory chip and CPU involved in the process? Imagine "viral" taken to the next step. You post a video to Facebook, your friends can stream it from my phone, their friends can stream it from your friend's phones and onward from that (a "circulate" button on Facebook that eliminates indefinite archiving). The idea would essentially provide a hypervisor for Facebook. The idea is far-fetched in terms of network availability to mobile phones, but that will likely be changing very soon. A safer suggestion for the Hypervisor community is the Fully Virtualized Datacenter. A new idea that likely will dominate the industry for the next decade and a half. Ken Won wrote "The link between servers and storage is the network. What we are discovering today is that virtualizing the network eliminates manual processes, costs, errors, and inefficiencies that are caused by the demands of virtualized servers and storage [16]. To support virtualized environments, the network needs to change from being a static data interconnect to a dynamic resource that can be automatically provisioned along with VMs and virtualized storage. New products are being brought to market to support this capability [17]. This is just one idea behind the "fully virtualized data center" that defines its usability. Universal improvements that we have yet to see. The future of hypervisors is exciting, and remarkably under the radar for most people. Eventually it will dominate everything that isn't gaming with an efficiency that would blow the minds of the initial innovators who brought the idea to life back in the mid 1960's. Hats off to them.

5.REFERENCES

1. “How did the term “hypervisor” come into use?” http://softwareengineering.stackexchange.com/questions/196405/how-did-the-term-hypervisor-come-into-use
2. Rui Natário. “Full Virtualization Explained” http://networksandservers.blogspot.com/2011/11/full-virtualization-explained.html
3. Gil Neiger, Amy Santoni, Felix Leung, Dion Rodgers, Rich Uhlig. Intel Virtualization Technology: Software-only virtualization with the IA-32 and Itanium architectures, Intel Technology Journal, Volume 10 Issue 03, August 2006.
4. "Xen dom0 support in Lucid - Kernel team discussions - ArchiveOrange". Web.archiveorange.com. Retrieved January 22, 2014.
5. Matthew Palmer (July 13, 2012). "vSwitch the New Battleground for Network Virtualization". Sdncentral.com. Retrieved August 25, 2016.
6. Christian Hammond (January 26, 2016). "A Tribute to VMware Workstation, Fusion, and Hosted UI". ChipLog. Retrieved August 25, 2016.
7. "Red Hat Advances Virtualization Leadership with Qumranet, Inc. Acquisition". Red Hat. 4 September 2008. Retrieved 16 June 2015.
8. "Linux kernel 2.6.20, Section 2.2. Virtualization support through KVM". kernelnewbies.org. 2007-02-05. Retrieved 2014-06-16.
9. Deepak Prasad, “Comparison Type 1 vs Type 2 Hypervisor”. July 01, 2014. http://www.golinuxhub.com/2014/07/comparison-type-1-vs-type-2-hypervisor.html
10. Matterson, S. (2013, August 8). 10 things you shouldn't virtualize - TechRepublic. Retrieved December 12, 2016, from http://www.techrepublic.com/blog/10-things/10-things-you-shouldnt-virtualize/
11. Danielle Ruest. “Virtualization hypervisor comparison: Type 1 vs. Type 2 hypervisors” techtarget. September 2010. http://searchservervirtualization.techtarget.com/tip/Virtualization-hypervisor-comparison-Type-1-vs-Type-2-hypervisors
12. Won, Ken. Implementing a Fully-Virtualized Data Center. Retrieved December 12, 2016, http://storypr.com/images/Articles/Fully_Virtualized_Data_Center.pdf
13. Jones, Penny. (2013, October 14). The future of virtualization. Retrieved December 10, 2016, http://www.datacenterdynamics.com/content-tracks/servers-storage/the-future-of-virtualization/82714.fullarticle
14. Gartner, Susan M. (2016, May 12). Worldwide Server Virtualization Market is Reaching Its Peak. Newsroom. Retrieved December 6, 2016 http://www.gartner.com/newsroom/id/3315817
15. Galland, James, B. (2016, September 1). System and method for nested hypervisors and layer 2 interconnection. The Faction Group LLC. Retrieved November 5, 2016, https://www.google.com/patents/US20160253198
16. Load Balancing and Smart NIC. Retrieved November 2, 2016, from http://www.ethernitynet.com/solutions/load-balancing-and-smart-nic/
17. Pratt, I.. WhyXen_Brochure. Xen.org. Retrieved November 6, 2016, from http://www-archive.xenproject.org/files/Marketing/WhyXen_Brochure.pdf