CHAPTER 1

Understanding Virtualization

IN THIS CHAPTER

- Understanding the different categories of virtualization
- Understanding early session virtualization
- Changing focus in corporate datacenters
- Examining the cloud and cloud services
- Meeting the needs of mobile workers
- Using virtualization to address today's challenges

Virtualization is a somewhat broad term that has gotten even broader over time as users, organizations, and the technologies they use have evolved. This chapter starts at the beginning of the virtualization journey—my journey anyway. I share my personal experiences to not only highlight the dramatic changes that have taken place, but also to demonstrate the many features of today's technology that still echo the ideas from over three decades ago. I discuss the ways the technology has grown, shifted, and made its way into nearly every organization and almost every aspect of our digital world. This chapter describes the main trends in virtualization over the past 30 years and provides a good background for the rest of the book, which dives deeper into Microsoft's products in each area of virtualization.

WHAT IS VIRTUALIZATION?

It's important to start by defining the term virtualization. It means different things to different people when thinking about computing. For the purposes of this book, you can think of *virtualization* as breaking the bonds between different aspects of the computing environment, abstracting a certain feature or functionality from other parts. This abstraction and breaking of tight bonds provides great flexibility in system design and enables many of the current capabilities that are the focus of IT and this book.

Over time the "virtualization" tag was applied to many other technologies that had been around for some time, because they also broke those tight couplings and abstracted concepts. Over the next few pages I introduce the major types of virtualization, which are explored in detail throughout the book.

When the word *virtualization* is used without qualification, many people think of machine virtualization, which is the easiest to understand. With machine virtualization the abstraction occurs between the operating system and the hardware via a hypervisor. The *hypervisor* divides up the physical resources of the server, such as processor and memory, into *virtual machines*. These virtual (synthetic) machines have virtual hard disks, network adapters, and other system resources independent of the physical hardware. This means you can typically move a virtual machine fairly easily between different physical machines, as long as they use the same hypervisor. If you try to move a system drive from one physical computer and put it in another, it's unlikely to work well, if at all, because of differences in hardware configuration. In addition, by creating several virtual machines on one physical computer, you can run multiple instances of the operating system on a single server, gaining higher utilization as hardware is consolidated, an idea I expand on later in this chapter.

In *presentation virtualization*, also called *session virtualization*, the user session is abstracted from the local device and runs on a remote server with connections from multiple users. Only the screen updates are sent to each user's local device, while all the computing actually takes place on the remote server. In other words, the presentation of the session is abstracted from where the actual computation takes place. Terminal Services and Citrix XenApp are examples of session virtualization solutions.

Technologies that enable users to use many different devices but with the same data and environment configuration have also gained the virtualization stamp. Users of previous Windows operating systems will know of Folder Redirection and Roaming Profiles. Later in the book you learn about other, more advanced technologies, particularly for the virtualization of user settings. Here again, the user data

and settings are abstracted from the underlying computer on which they are typically hard linked into the operating system.

One relatively new technology is *application virtualization*, which enables the decoupling of an application from the operating system. Traditionally, in order to use an application it typically had to be installed on the user's computer, adding components onto the operating system, updating settings containers, and writing data to the local disk. With application virtualization, application code is downloaded from a remote site and runs locally on a computer without requiring any changes to the operating system, so it has zero footprint. Note that this differs from session virtualization in that the computation is on the user's device, not on the remote server. The Microsoft application virtualization technology is App-V.

Throughout this book I describe the technologies that implement these and other categories of virtualization, including how they are used, when they should be used, and how to build the right IT infrastructure using the appropriate virtualization technology. To provide some context, the following section looks at changes in the industry over the last 30 years as I've experienced them. This is my personal view, not a traditional textbook history of computers, which you can find elsewhere. It reflects what I've seen in my years of consulting and acting as a trusted advisor for enterprises of various sizes and stripes, and provides some insight into what lies ahead.

The Dawn of Virtualization

When I was about eight years old I got my first computer, a ZX Spectrum with 48KB of memory, which connected to the television, and software (games mostly) on cassette tapes. I still have it on the wall in my office as a reminder of where my love of computers started. I played around with the BASIC language that came with it, creating epic code such as the following and feeling very proud when I would enter it on machines in stores:

```
10 PRINT "JOHN IS GREAT"
20 GOTO 10
```

Over time I moved on to a Dragon 32 that used cartridges, a new Spectrum with 128KB of memory and built-in tape drive, called the ZX Spectrum 128 +2, and a Commodore Amiga. Then one day my dad brought home a PC—I think it was a 286 with MS-DOS and 5.25-inch disks. When the technology evolved to the point of 386 computers and acquired larger internal disks, we upgraded our machine, installed Windows, and I started to play around with the C programming language and later with Java.

▶ For American readers, the ZX Spectrum was similar to a Commodore 64. There were many schoolyard arguments over which one was better.

When I was 18 years old I got a job at Logica, which at the time was a large systems house. I worked in the Financial Services division and was hired as the VAX/VMS systems administrator. I had no clue what that meant, but they said they would train me and pay some of my tuition while I worked toward my degree. The position sounded amazingly advanced, and as I walked into my first day at work I had visions of futuristic computing devices that would make my home machine look like junk. Unfortunately, instead of some mind-controlled holographic projection computer, I saw an old-looking console screen with green text on it.

As I would later find out, this device (a VT220) was just a dumb terminal that allowed keyboard entry to be sent to a VAX/VMS box that sat in the basement (where I spent a large part of my early "systems management" duties changing backup tapes and collecting printouts to deliver to developers on the team). The VAX/VMS server had all the actual computing power, memory, storage, and network connectivity. Everyone in the team shared the servers and had their own session on this shared computer, which used time-sharing of the computer's resources, specifically the CPU. This was very different from my experience up to that point, whereby all the computation was performed on my actual device. For these large enterprise applications, however, it made sense to share the computer power; so there were multiple instances of our application running on the same physical server, each in its own space. There were other mainframe devices from IBM that actually created virtual environments that could act as separate environments. Windows for Workgroups–based PCs were introduced into our team and certain types of workload, such as document creation, moved to the GUI-based Windows device.

▶ Session virtualization!

▶ Machine virtualization!

I discovered Windows NT by accident one day when I pressed Ctrl+Alt+Del to reboot and a security dialog was displayed instead. This Windows NT box was used as a file server for data the team created on the machines. I started investigating and learning the Windows NT technology, which is when I started the `SavillTech.com` and `ntfaq.com` sites to answer frequently asked questions (FAQs).

In order to really test and investigate how to perform certain actions, I needed multiple Windows NT servers for different types of server role and workload, and I needed a client. As a result, at one point I had six desktop computers running either NT Server or NT Workstation, one for each role. I found some nice drive bays that enabled me to switch out the hard drives easily, so I could change the particular operating system a physical box was running. Sometimes I could dual boot (choose between two operating systems on one disk), but these servers were not very active. Most of the time the CPU was barely used, the memory wasn't doing that much, and the busiest part was the disk.

▶ This one physical box for each operating system instance is exactly the same process that organizations followed until a few years ago.

A turning point in my long relationship with computers occurred while watching a presenter at a conference. The exact date and conference elude me, but I'm pretty sure it was in fact a Microsoft conference, which strikes me as ironic now. I remember little from the session, but one thing stuck in my mind and eclipsed everything else that day. The presenter had one machine on which he was running multiple operating systems simultaneously. This was a foreign concept to me, and I remember getting very excited about the possibility of actually fitting my legs under my crowded desk. My electricity bill might become more manageable as well! I strained my eyes to see what was being used and discovered it was VMware Workstation.

The next day at work I researched this machine virtualization technology that enabled one physical box to be carved into multiple virtual environments. The VMware Workstation software was installed on top of Windows as an application that allocated resources from the actual Windows installation on the physical box. Known as a *type 2 hypervisor*, this solution did not directly sit on the actual hardware but rather worked through another operating system. The downside of this type 2 hypervisor was some performance loss, as all operations had to run through the host operating system, but for my testing purposes it worked great. At home, I quickly went from six boxes to three boxes, as shown in Figure 1-1. I then increased the actual number of operating system instances, as they were now easy to create and didn't cost me anything extra because I had capacity on my machines. I just threw in some extra memory (which I could afford by selling the other three computers) when I needed more VMs, as typically that was my limiting factor.

By this time I had changed jobs and now worked at Deutsche Bank, where I was an architect creating a new middleware system for connecting the various financial systems in the U.K. The system transformed messages from the native format of each system to a common generic format to enable easy routing. This was implemented on Sun Solaris using an Oracle database, and I wrote the front end in Java. Interestingly, there were some Windows-based financial applications that accessed huge amounts of data, which created heavy data traffic to and from the database. Such applications performed very poorly if they were run on a desktop outside of the datacenter because of network constraints. To solve this, the applications ran on a Windows Server that was located in the datacenter with Terminal Services installed. Users accessed an application from either a Remote Desktop Protocol (RDP) client on the Windows Server itself or a special thin RDP client. This was essentially a glorified version of my first VT220, but the green text was replaced with a color bitmap display; keyboard-only input was now keyboard and mouse; and communication was now via IP (Internet Protocol) instead of LAT (Local Area Transport). Nevertheless, it was exactly the same concept: a dumb client with all the computation performed on

Remember that only the screen updates (bitmaps) and the keyboard/mouse commands are sent over the link, so the actual data never travels across the network.

a server that was shared by many users. It was back to session virtualization! I was also informed that because the data was sensitive, this session virtualization was preferred, because running the application in the datacenter meant the data never left there.

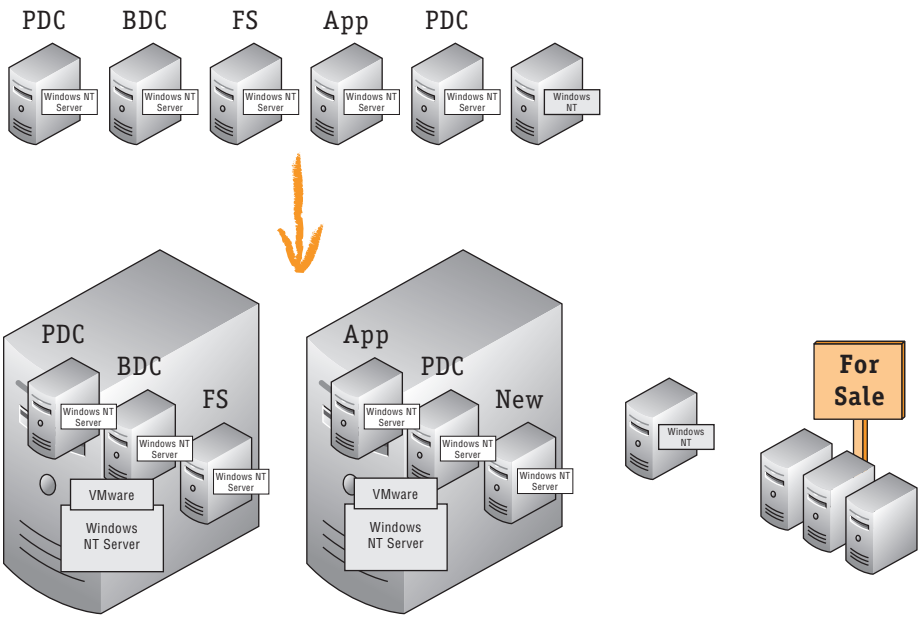

FIGURE 1-1: Using a type 2 hypervisor, I was able to move from six machines to three, and increased the number of operating system instances I could run. Note that the two PDCs reflect a second domain.

Time passed and Microsoft purchased Connectix, which made an alternative solution to VMware Workstation, Virtual PC, giving Microsoft its own machine virtualization solution. Virtual PC has both a desktop and a server version, Virtual Server, but both offerings are still type 2 hypervisors running on top of Windows. In the meantime, VMware released ESX, which is a *type 1 hypervisor*, meaning the hypervisor runs directly on the server hardware. The performance of a virtual machine running on such a hypervisor is nearly the same as running on the physical server. I actually used this technology for a while and it was very powerful; Microsoft didn't really have anything comparable. This all changed with the release of Windows Server 2008, which included Hyper-V, a type 1 hypervisor. What we think of as Windows Server actually sits on top of the hypervisor to act as a management partition. As you will see in Chapter 8, which focuses on Hyper-V, it has gone from strength to strength—to the point that it is now considered one of the top hypervisors in the industry.

My lab still consists of three servers but they are a lot bigger. They run a mix of Windows Server 2008 R2 and Windows Server 2012 Hyper-V, and I now have around 35 virtual machines running! One change I recently made was to my e-mail. For a period of time I experimented with hosting e-mail on Exchange in my lab, but I never took the time to ensure I was backing up the data regularly or to enable new services. I've now moved over to the Office 365–hosted Exchange, Lync, and SharePoint solution, as I am also working on some other projects for which I want SharePoint and improved messaging for collaboration. Quite honestly, I didn't want to maintain an internal SharePoint solution and worry about backing up important data.

In short, I have seen dramatic change over 30 years of working and playing with computers. As much as the technology has changed, however, many features still echo the ideas that were current when I first started, such as session virtualization and consolidation of workloads.

The Evolution of the Datacenter and the Cloud

A lot of what I've described so far mirrors the evolution of corporate datacenters and IT departments around the world. This journey for many companies starts with each operating system running on its own physical box—a number of servers for the domain controllers, a number of file servers, mail servers, you get the idea. To decide which server to buy for a new application, the highest possible peak load required for that application is used. For example, a Friday night batch might be used to decide the level of hardware needed. Also considered is the future growth of the server's load over its lifetime, typically 4–6 years, meaning a company usually purchases a server that will run at about 10 percent CPU capacity and maybe use only half the amount of memory available just to cover those few peak times and future growth *if* everything goes according to plan. This can be a large financial outlay that wastes rack space and power. The bigger, more powerful servers also generate more heat, which in turn requires more cooling and power, which also translates into greater cost. Strict processes are required to deploy anything new because of the associated costs to procure the hardware.

Most datacenters use large cabinets that hold a number of servers that slot in horizontally. These servers typically take up 1, 2, or 4 units of space in the rack, so the bigger the servers the more space is occupied in the rack, therefore requiring more racks.

Backups are performed nightly to tape, which is either kept in a closet or, for organizations concerned about the loss of a site, sent offsite and rotated on a regular cycle (maybe every six weeks) to enable restoration from different days in case corruption or deletion is not immediately noticed. Storage is also very expensive, so applications are carefully designed in terms of data use.

Features such as clustering became more popular because they enabled certain services and applications to be *highly available*—a service can automatically move

to another server that is part of the same cluster should a server fail, ensuring the service is always available to users. The cluster solutions rely on shared storage, which can be expensive, but the high availability gained for critical services justifies the external storage solutions, which also offer other capabilities, such as higher data integrity and easy duplication and backup of data.

Some organizations need the ability to continue operations even in the event of complete site loss—for example, a flood or other natural disaster. In these scenarios, organizations opt for a second datacenter in an alternate location that can run their critical systems. These disaster recovery (DR) datacenters are typically categorized as follows:

- **Hot:** A hot site is a second location that replicates almost exactly the primary datacenter and has near real-time replicas of the data. If the primary location fails, a hot DR site is up and running quickly. This option requires complex networking and expensive technologies, so it is typically reserved for the most critical systems, such as those used by government and financial organizations.
- **Warm:** A warm site also has alternate systems available, but they do not replicate in real time; so, there may be some lag in data transfer. Manual processes are required to get up and running on a warm site, making it much slower to start performing than a hot site, but costs are typically significantly less than those of a hot site.
- **Cold:** A cold site is a location that may have some servers but they typically don't match the production site hardware. Nor does this option provide data replication or easy access to backups. It provides an alternate location, but it requires a significant setup effort before it can actually be used, meaning there is a long lag time between a failure and when the organization is up and running again. The exact time depends on number of systems, size of backups to be restored, and complexity of the environment.

For the warm and cold options, I have seen many companies leverage third-party services that effectively rent out servers and facilities in the event of a disaster, based on a retainer fee. These services enable companies to avoid maintaining a second location and complete set of servers but still give them a disaster recovery service should it be needed—at significantly lower cost than running their own non-shared DR site.

MAKING SURE YOUR DISASTER RECOVERY IS USABLE IN A DISASTER

Failing to prepare is preparing to fail.

—John Wooden, head coach of the UCLA Bruins, 10-time National Championship winners

For all the DR options, because the operating system backups are hardware specific, DR sites must have replica hardware of the primary site (unless virtualization is used), which is hard to maintain. Keep in mind, this is a huge expense for a scenario that might never occur. In addition, if DR locations are used, it's vital to test the failover process periodically. I've worked with a number of companies that invested in DR capabilities but rarely tested them; and when they actually had to use it or finally performed a test, they realized they had forgotten to add a replica of some key workload on which everything else depended, making the DR site basically useless.

A test every six months is a good middle ground. Many organizations test by running live on the DR site for an entire week before moving back to their primary location. This requires a lot of planning and work to reverse the replication from the DR to production. Another option is to have the capability at the DR location to run all required processes in a separate test network, enabling testing of the DR systems without affecting production. Even if a third-party service is used for the DR service, it's important to ensure that a periodic test is included in the service.

As IT systems become increasingly important to organizations and the number of IT systems grows, so does the complexity and amount of work required to maintain those systems. Many companies begin to suffer from *server sprawl,* where a disproportionate number of servers are available relative to the actual resources needed, which results in huge amounts of IT administration overhead. Organizations begin to adopt technologies to help manage all the physical servers and operating system instances, but it quickly becomes apparent that these organizations are spending huge amounts of money on hardware and power for servers that are greatly underutilized.

Power is also getting a lot more expensive and people are becoming concerned about the ecological costs of growth. When it became impossible to ignore that the

planet was getting a little toasty, there was a large wave of interest in being more energy efficient. Virtualization is the main enabler for more efficient datacenters that better utilize the available resources, reducing the hardware footprint and saving money and power.

Efficiency has also improved through the use of energy-saving technological advancements, such as processors that require less power. Operating systems also made a huge leap forward with the ability to turn off idle cores on multi-core processors, condensing the need for 10 physical servers into one using virtualization for the operating systems. Today, many organizations are making this move from physical server to virtual server for many of their workloads. (Up to this point I have been the Ghost of IT Past; now I'm shifting to the Ghost of IT Present.)

The move from physical to virtual is a big endeavor. Careful analysis of all the existing servers must be performed to understand their actual resource usage in terms of memory, processor, disk, and network—including average resource utilization versus peak loads, and when those peaks occur. Only then can plans be made as to where to co-locate various existing systems when virtualized onto shared physical hosts. For example, suppose you have two physical servers, one peaking on a Friday night, in terms of resource utilization, and the other peaking on Monday morning. In this case, you could place those two systems on the same virtualization host because their peaks occur at different times. That way, you only need to plan out the actual resource of the virtualization host to handle the average load of one server and one peak. This same principle applies to the entire environment, in reality. Typically 10 to 20 operating system instances will end up on a single host. Hosts can then be clustered together to enable high availability for the virtual environment and even enable the movement of virtual machines around hosts based on resource requirement changes. In other words, if a virtual machine gets very busy and is causing resource contention with other virtual machines on a host, then the virtual machine can be moved to a different host that is less utilized.

▶ The Microsoft Assessment and Planning (MAP) Toolkit helps perform this analysis and even identifies the best placement of operating system instances on physical hosts when virtualized. The best news is it's free!

Virtualization is not without its challenges, though. Yes, your hardware is better utilized. Yes, you save money on hardware, power, and even licensing. Yes, your DR is far simpler because now the operating systems are virtualized, meaning the actual hardware does not matter because the operating systems are running inside the VM. Therefore, you can use one type of server in your datacenter and something completely different at the DR location; you just need to use the same hypervisor. Yes, you can provision new operating systems in virtual machines very quickly because you don't have to wait for a physical server to arrive, and you can create a new operating system instance by deploying a template (a prepared operating system installation ready for duplication). Yes, it can even make applications that don't natively have any kind of

high availability highly available, by making the VM highly available at the hypervisor level through clustering technologies. Yes, it sounds fantastic, and machine virtualization absolutely is a huge asset to organizations or it wouldn't have been adopted so readily, but nothing is ever perfect.

One common side effect of virtualization is *virtualization sprawl*, or the uncontrolled creation of virtual machines in an organization leading to an environment that's hard to manage or audit, similar to server sprawl. Suddenly there is no direct cost correlation between new operating system instances, which are now virtual machines and not physical servers that have a purchase price. It can become easy to just create new virtual machines and not track their usage, resulting in the maintenance of operating systems that are no longer used. There are technologies to help mitigate this issue, such as charging business units based on the number of virtual machines they have or perhaps computer hours used. Another option is to assign business units a quota of virtual instances and then implement tracking to identify unused virtual environments that can be archived.

Aside from the tendency to sprawl, there is a bigger challenge: Virtualization has not helped reduce the management of the IT infrastructure; it has actually made it harder. Where previously I had 1,000 physical servers, each running one operating system and an application, I now have 1,000 virtual machines, each with its own operating system that needs all the same patching and maintenance as the operating systems on those physical boxes needed. I have not cut down on the amount of management work required. Now, I also have to manage the virtualization environment, and because I've consolidated resources, I now have to carefully track the resource utilization of processors, memory, disks, and the network to ensure that the virtual machines are sufficiently allocated on the shared infrastructure. Additionally, before I had one egg in each basket, metaphorically speaking, so if I dropped one basket I lost one egg. With virtualization, I now have a lot of eggs in each basket, so dropping a basket has huge ramifications. Therefore, extra planning is required to ensure the resiliency and availability of the virtualization host environment, including networking and storage.

Some companies have shifted to management tools that enable automation because of the huge number of operating systems their IT departments have to manage. Remember, the more you can automate a process, the easier it will be to manage and the more consistent the result will be. There are a number of great Microsoft tools to better manage an infrastructure that uses virtualization. I have listed a few of them here, and I cover them in far greater detail in Chapters 9 and 10.

- System Center Configuration Manager (SCCM) can easily deploy patches to thousands of servers in a staged and controlled manner. It also provides the capability to easily set maintenance windows for different groups of servers.

- Virtualization management solutions, such as System Center Virtual Machine Manager (SCVMM), can simplify deployment in a virtual environment through the use of templates, which enable a company to define server operating system instances without the need for human intervention—other than giving the new VM a name.
- System Center Orchestrator, and other tools, use run books to automate manual tasks performed on many different systems.
- PowerShell enables the complete management of servers and applications from a command and scripting environment and remote management on many servers simultaneously. If you have never looked at PowerShell, start now. PowerShell is very much at the heart of Windows 8.

A run book is a set of defined actions for tasks performed on one or more systems, such as regular maintenance activities or exception processes.

UNDERSTANDING THE TYPES OF CLOUD AND CLOUD SERVICES

Today's organizations want the freedom to focus on the application or service rather than all the individual operating system instances running inside virtual machines. Business units and users need to be able to provision their own virtual environment, or *private cloud*, within the structure predefined by the IT group, using simple self-service portals. This shift is the next step in the IT journey—from physical to virtual and now to private cloud.

CROSSREF This shift to the private cloud is covered in detail in Chapter 10.

The private cloud builds on virtualization but focuses on providing services to businesses in a highly scalable but manageable manner with infrastructure components taking care of the creation and maintenance of the underlying virtual machines and operating systems. The private cloud enables a company to focus on the service being provided, which could be a multi-tiered service consisting of numerous web servers interacting with clients, then a middleware tier performing processing, then a back-end database tier. With a private cloud, a complete instance of the entire service can be deployed with the click of a button; and the actual servicing of the environment is transparent to the operation of the service.

If certain aspects of the service are becoming highly utilized, then that tier can easily be scaled out by merely requesting additional instances of that tier; the infrastructure handles all the work of creating additional virtual machines, installing the correct code, and adding to any load balancing.

To many business users and executives, the idea of the private cloud is fantastic. To their IT departments, the private cloud is terrifying. Many of my clients have expressed concern that business users can just create their own VMs—and even worse, *many* VMs as part of a single service deployment. In other words, they fear they will be overrun. However, it's important to realize that the private cloud is a complete solution comprised of many parts. It has the capabilities to model services, deploy virtual environments and install the needed applications, and maintain them. It has an easy-to-use, self-service, end user interface that relies on templates; but it also has powerful capabilities that enable IT departments to model a cloud of resources and features to which their business users are confined and to define quotas limiting the amount of resources each group or user consumes. *Showback* and *chargeback* capabilities can be used to track and charge (respectively) business units for what they actually use, rather than creating huge numbers of virtual environments and services. The IT department is still involved, but it does its work up front, proactively defining groups of resources as clouds and then assigning them to groups of users.

The looming private cloud is needed. Traditional virtualization enables the quick creation of a virtual environment, but it's a manual exercise for the busy IT administrator; and fulfilling a user request for a virtual machine can actually take several weeks. Users won't wait this long if there is another option (which I'm getting to). As mentioned earlier, with the private cloud the IT work is done up front, so users can create new environments using self-service interfaces in minutes; and, if needed, workflows can still be used for manager approval or other sign-off actions. So what is this other option? The public cloud.

The *public cloud* is a huge factor for nearly every organization today. It offers various services that are hosted in the provider datacenters and accessed over the Internet by an organization. This means the organization requires no infrastructure, and typically just pays for what it uses. It would be incorrect to think of the public cloud as only appropriate for business users who are unhappy with their local IT departments. The public cloud has extensive capabilities and covers multiple scenarios for which an organization's IT infrastructure is not the right solution, and many organizations will make use of both private and public clouds.

CROSSREF Chapter 14 covers public clouds and their services in greater detail.

The following material covers the three types of public cloud service that are typical today: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS):

- **IaaS:** You can think of IaaS as a virtual machine in the cloud. The provider has a virtual environment and you purchase virtual machine instances, within which you manage the operating system, the patching, the data, and the applications. An example of IaaS is Amazon's Elastic Compute Cloud (EC2), which enables an organization to run operating systems inside Amazon's virtual environment.
- **PaaS:** PaaS provides a framework on which custom applications can be run. Organizations can focus on writing the best application possible within the guidelines of the platform's capabilities; everything else is taken care of. There are no worries about patching operating systems, updating frameworks, backing up SQL databases, or configuring high availability. After providing the application, the organization pays only for the resources used. Windows Azure is the classic example of a PaaS.
- **SaaS:** SaaS offers the ultimate public cloud service in terms of lowest maintenance. It is a complete solution provided by the vendor. Organizations don't need to write or maintain anything; their only responsibility is to configure who should be allowed to use the software. A commercial example of SaaS is Hotmail, the popular online messaging service. An enterprise example is Office 365, which provides a cloud-hosted Exchange, SharePoint, and Lync service—with everything accessed over the Internet. No application or operating system management is required by the customer.

Figure 1-2 clearly highlights the differences between these three services. The shaded areas depict the management areas for which the customer is responsible. The on-premise solution is self-explanatory; the organization must manage all aspects of its own environment.

Moving to IaaS basically provides the capability to host virtual machines. As shown in Figure 1-2, the IaaS provider is responsible for networking, storage, server, and the virtualization layer; you are responsible for all aspects of the operating system within the VM, the middleware, the runtime, data, and of course the application. While it may seem like the IaaS provides the most flexibility, the trade-off is the amount of management still required; however, many organizations test the water by

first using IaaS before moving on to reduce their management efforts and gain the benefits offered with PaaS and SaaS.

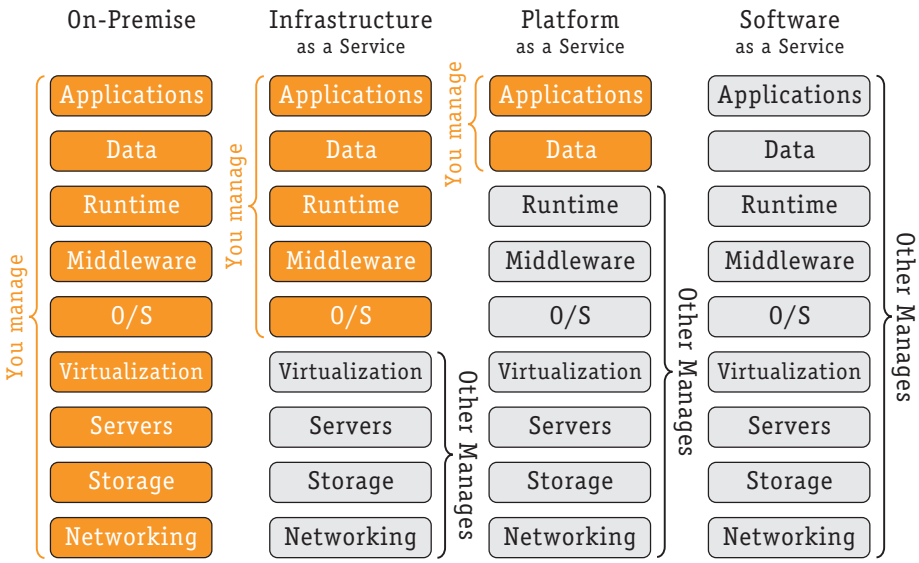

FIGURE 1-2: The nine main management areas—from the network to the application—and the parts you have to manage for each type of service.

Moving to Platform as a Service dramatically changes the amount of management your organization needs to perform. With PaaS, you only have to concern yourself with the application you manage and its data, leaving everything else to the PaaS provider. Note that although you manage the data, the provider typically still provides services to actually replicate and protect the data.

Finally, when you move to using Software as a Service you are no longer responsible for managing anything in terms of infrastructure—you just use the cloud-based software. Not every system can use SaaS, as organizations have their own custom code, but the goal for many organizations where software cannot be provided by SaaS is a combination of PaaS and IaaS; and there will always be some on-premise systems.

SHIFTING TECHNOLOGICAL PARADIGMS

Of course the emergence of the cloud wasn't driven solely by organizations needing more manageable virtualization solutions. The increasing desire of workers for flexibility in terms of where and how they perform their computing tasks has had a tremendous influence. The line between work life and personal life has blurred as

Tethering a mobile device to a laptop enables the laptop to use the mobile device's data connection, thereby connecting the laptop to the Internet.

a natural consequence of the many different devices available to us now. Consider how quickly the cell phone morphed into a "smartphone," capable of handling many aspects of our busy lives, including checking e-mail, managing your calendar, linking to social media, and keeping you informed with the latest news. With a great computer at home with the latest operating system and broadband connectivity to the Internet, slate devices with Wi-Fi, and mobile devices with very fast data capabilities that can be tethered to laptops, users can work anywhere—at the office, at home, or even at the local Starbucks. Technology needs to enable this kind of flexibility that users demand.

Understanding the Needs of the Mobile Worker

Notice "device," not PC! There was an earlier movement to Bring Your Own PC (BYOPC), but this has changed now to any kind of device (BYOD), including slates such as the iPad.

Just being connected is not enough for most mobile workers. In order to be productive, they need the right kind of secure connectivity to corporate resources and it must be available from many different locations. Additionally, the mobile user may need to access corporate resources using many different types of device, including some with newer or different operating systems than what they have at work, including those running iOS or Android with no ability to run native Windows applications. iPad owners, for example, want to be able to connect it to the corporate e-mail and use it at work. In the past, organizations could simply refuse to accommodate the few workers who wanted outside access; however, today it's the senior executives at organizations bringing in the new wave of slate devices and requiring the IT department to find a way to make them work. This shift in power, where consumers have the latest technologies and gradually force their workplaces to integrate them, is referred to as the *consumerization of IT*.

CROSSREF You can read more about enabling secure connections for mobile workers in Chapter 12.

What Consumerization Means to an Organization

The consumerization of IT is changing the way organizations manage their resources. In fact, instead of providing users with computers, many companies are allocating an annual computer budget for users to buy their own machines. But beyond changes in accounting, users utilizing their own devices for work presents a huge challenge

from an IT perspective. Some devices may not be running corporate installations of Windows and may not even be capable of running some corporate applications. The iPad, for example, runs iOS, which is really a mobile device operating system with limited ability to run applications locally. Even when the device is capable, problems may still persist. Licensing would not work, users often don't want corporate applications installed on their devices, and many applications integrate with Active Directory, which wouldn't be present on a noncorporate device. Clearly, another solution is needed to enable corporate application and data access on these noncorporate assets.

One of the biggest concerns is security, because these devices may not have the latest anti-virus solutions or firewalls installed. A user's personal device is unlikely to have the necessary encryption to protect corporate data, so users are limited in terms of what they can store on noncorporate assets. If you look at the lack of security on the average home machine, you would likely not want to let it near your corporate network! Connectivity needs to be enabled for these mobile users while still maintaining the security and health of the corporate environment.

Users working on personal devices also present legal problems. When a corporate device is lost, the organization can often perform a remote wipe operation to delete everything. Such a solution isn't going to be very popular with a user if it means deleting baby's first steps. Policies need to be in place around using personal devices for corporate purposes. For example, such a policy could specify appropriate access to corporate e-mail on an Exchange server using a noncorporate mobile device. To enable access, the device could be required to accept the security policy set by Exchange ActiveSync, which might include requiring a PIN, and to have lockout and wipe settings.

The challenges presented by the consumerization of IT are significant and varied, but there are solutions and they require a bit of a return to our technology roots.

EMBRACING THE RETURN OF THE DUMB TERMINAL

Virtual Desktop Infrastructure (VDI)—the desktop equivalent of the cloud—and session virtualization can often provide the solution to some of the problems presented by mobile workers. VDI enables users to remotely connect to a client operating system running in a virtual environment within the corporate datacenter.

Session virtualization, on the other hand, is an often overlooked option that can be a better fit than VDI for many scenarios. While VDI provides each user with his own client operating system on a virtual machine, session virtualization provides the same user experience but provides a session on a shared server operating system. VDI is frequently seen as a cure-all for everything from poor desktop management to solving application compatibility problems. Although that isn't the case, it's still a great solution when used in the right ways.

> CROSSREF Chapter 11 describes the VDI in detail, and Chapter 7 covers session virtualization.

Regardless of which option you use, you can provide a Windows environment for outside users who are connected to the corporate infrastructure that can run corporate applications and access corporate data, all while running within the datacenter. The user accesses this Windows environment through remote desktop protocols using a client application that could be running on an iPad, a mobile phone, a home machine, or a laptop. The device doesn't matter because it is just serving as a dumb terminal (like my old VT220). The device displays the Windows desktop and applications, but the actual work is being done on a back-end server. Therefore, the user's device does not have to be part of the domain. It doesn't even have to be capable of running Windows applications; it just needs a client that can communicate the Windows environment in the datacenter using the remote protocol. The concerns related to data on a device are also resolved using VDI and session virtualization because no corporate data is stored locally on the client device—it's all in the datacenter with the user's session, which means there are no concerns about lost corporate data and wiping devices if a user loses his laptop.

While changes to the datacenter have been more revolutionary than evolutionary, the world of the desktop has not exactly stood still. Virtualization is also a critical aspect of the desktop today, especially in terms of meeting the new flexibility requirements governing how users want, and often need, to work, and as organizations replace older Windows versions with Windows 7 or Windows 8. The main areas of focus for the desktop relate to client virtualization, user data and settings virtualization, and application virtualization, all of which are discussed in the coming chapters, although many other types of virtualization can also come into play when architecting the best desktop infrastructure and supporting multiple client devices.

SUMMARY

The brief tour of virtualization you took in this chapter should have demonstrated that virtualization solves many problems and enables many new ways of working, but only if it is well architected and implemented. If not, it can cause more work and introduce new problems. Embracing the consumerization of IT and offering mobile users a productive environment is achieved through virtualization; there really is no other solution. Through technologies such as VDI, session virtualization, and even public cloud services, users on any device and in any connected location can have a functionally rich environment within which to work. I often find myself on a device that is acting as nothing more than a much better version of my VT220 dumb terminal, and if I were the average user, I would likely have no idea that the operating system environment within which I was working wasn't running on the device in my hand.

The rest of this book walks you through the major types of virtualization that are available, how they can be used, and how they *should* be used. Chapter 13 covers the different types of on-premise virtualization, putting them into context so you can architect the right infrastructure for your organization. Ultimately, most organizations will have a mix of on-premise solutions and public cloud solutions in order to make the best use of all the available technologies—and at a price they can afford.