**The lecture 4**

**Overview of cloud computing**

Cloud computing provides a modern alternative to the traditional on-premises datacenter. A public cloud vendor is completely responsible for hardware purchase and maintenance and provides a wide variety of platform services that you can use. You lease whatever hardware and software services you require on an as-needed basis, thereby converting what had been a capital expense for hardware purchase into an operational expense. It also allows you to lease access to hardware and software resources that would be too expensive to purchase. Although you are limited to the hardware provided by the cloud vendor, you only have to pay for it when you use it.

Cloud environments provide an online portal experience, making it easy for users to manage compute, storage, network, and application resources. For example, in the Azure portal, a user can create a virtual machine (VM) configuration specifying the following: the VM size (with regard to CPU, RAM, and local disks), the operating system, any predeployed software, the network configuration, and the location of the VM. The user then can deploy the VM based on that configuration and within a few minutes access the deployed VM. This quick deployment compares favorably with the previous mechanism for deploying a physical machine, which could take weeks just for the procurement cycle.

In addition to the public cloud just described, there are private and hybrid clouds. In a private cloud, you create a cloud environment in your own datacenter and provide self-service access to compute resources to users in your organization. This offers a simulation of a public cloud to your users, but you remain completely responsible for the purchase and maintenance of the hardware and software services you provide. A hybrid cloud integrates public and private clouds, allowing you to host workloads in the most appropriate location. For example, you could host a high-scale website in the public cloud and link it to a highly secure database hosted in your private cloud (or on-premises datacenter).

Microsoft provides support for public, private, and hybrid clouds. Microsoft Azure, the focus of this book, is a public cloud. Microsoft Azure Stack is an add-on to Windows Server 2016 that allows you to deploy many core Azure services in your own datacenter and provides a self-service portal experience to your users. You can integrate these into a hybrid cloud through the use of a virtual private network.

**Comparison of on-premises versus Azure**

With an on-premises infrastructure, you have complete control over the hardware and software that you deploy. Historically, this has led to hardware procurement decisions focused on scaling up; that is, purchasing a server with more cores to satisfy a performance need. With Azure, you can deploy only the hardware provided by Microsoft. This leads to a focus on scale-out through the deployment of additional compute nodes to satisfy a performance need. Although this has consequences for the design of an appropriate software architecture, there is now ample proof that the scale-out of commodity hardware is significantly more cost-effective than scale-up through expensive hardware.

Microsoft has deployed Azure datacenters in over 22 regions around the globe from Melbourne to Amsterdam and Sao Paulo to Singapore. Additionally, Microsoft has an arrangement with 21Vianet, making Azure available in two regions in China. Microsoft has also announced the deployment of Azure to another eight regions. Only the largest global enterprises are able to deploy datacenters in this manner, so using Azure makes it easy for enterprises of any size to deploy their services close to their customers, wherever they are in the world. And you can do that without ever leaving your office.

For startups, Azure allows you to start with very low cost and scale rapidly as you gain customers. You would not face a large up-front capital investment to create a new VM—or even several new VMs. The use of cloud computing fits well with the scale fast, fail fast model of startup growth.

Azure provides the flexibility to set up development and test configurations quickly. These deployments can be scripted, giving you the ability to spin up a development or test environment, do the testing, and spin it back down. This keeps the cost very low, and maintenance is almost nonexistent.

Another advantage of Azure is that you can try new versions of software without having to upgrade on-premises equipment. For example, if you want to see the ramifications of running your application against Microsoft SQL Server 2016 instead of Microsoft SQL Server 2014, you can create a SQL Server 2016 instance and run a copy of your services against the new database, all without having to allocate hardware and run wires. Or you can run on a VM with Microsoft Windows Server 2012 R2 instead of Microsoft Windows Server 2008 R2.

**Cloud offering**

Cloud computing usually is classified in three categories: SaaS, PaaS, and IaaS. However, as the cloud matures, the distinction among these is being eroded.

**SaaS: Software as a service**

SaaS is software that is centrally hosted and managed for the end customer. It usually is based on a multitenant architecture—a single version of the application is used for all customers. It can be scaled out to multiple instances to ensure the best performance in all locations. SaaS software typically is licensed through a monthly or annual subscription.

Microsoft Office 365 is a prototypical model of a SaaS offering. Subscribers pay a monthly or annual subscription fee, and they get Exchange as a Service (online and/or desktop Outlook), Storage as a Service (OneDrive), and the rest of the Microsoft Office Suite (online, the desktop version, or both). Subscribers are always provided the most recent version. This essentially allows you to have a Microsoft Exchange server without having to purchase a server and install and support Exchange—the Exchange server is managed for you, including software patches and updates. Compared to installing and upgrading Office every year, this is much less expensive and requires much less effort to keep updated.

Other examples of SaaS include Dropbox, WordPress, and Amazon Kindle.

**PaaS: Platform as a service**

With PaaS, you deploy your application into an application-hosting environment provided by the cloud service vendor. The developer provides the application, and the PaaS vendor provides the ability to deploy and run it. This frees developers from infrastructure management, allowing them to focus strictly on development.

Azure provides several PaaS compute offerings, including the Web Apps feature in Azure App Service and Azure Cloud Services (web and worker roles). In either case, developers have multiple ways to deploy their application without knowing anything about the nuts and bolts supporting it. Developers don’t have to create VMs, use Remote Desktop Protocol (RDP) to log into each one, and install the application. They just hit a button (or pretty close to it), and the tools provided by Microsoft provision the VMs and then deploy and install the application on them.

**IaaS: Infrastructure as a service**

An IaaS cloud vendor runs and manages server farms running virtualization software, enabling you to create VMs that run on the vendor’s infrastructure. Depending on the vendor, you can create a VM running Windows or Linux and install anything you want on it. Azure provides the ability to set up virtual networks, load balancers, and storage and to use many other services that run on its infrastructure. You don’t have control over the hardware or virtualization software, but you do have control over almost everything else. In fact, unlike PaaS, you are completely responsible for it.

Azure Virtual Machines, the Azure IaaS offering, is a popular choice when migrating services to Azure because it enables the “lift and shift” model for migration. You can configure a VM similar to the infrastructure currently running your services in your datacenter and migrate your software to the new VM. You might need to make tweaks, such as URLs to other services or storage, but many applications can be migrated in this manner.

Azure VM Scale Sets (VMSS) is built on top of Azure Virtual Machines and provides an easy way to deploy clusters of identical VMs. VMSS also supports autoscaling so that new VMs can be deployed automatically when required. This makes VMSS an ideal platform to host higher-level microservice compute clusters such as for Azure Service Fabric and the Azure Container Service.

**Azure services**

Azure includes many services in its cloud computing platform. Let’s talk about a few of them.

**Compute services** This includes the Azure Virtual Machines—both Linux and Windows, Cloud Services, App Services (Web Apps, Mobile Apps, Logic Apps, API Apps, and Function Apps), Batch (for large-scale parallel and batch compute jobs), RemoteApp, Service Fabric, and the Azure Container Service.

**Data services** This includes Microsoft Azure Storage (comprised of the Blob, Queue, Table, and Azure Files services), Azure SQL Database, DocumentDB, StorSimple, and the Redis Cache.

**Application services** This includes services that you can use to help build and operate your applications, such as Azure Active Directory (Azure AD), Service Bus for connecting distributed systems, HDInsight for processing big data, Azure Scheduler, and Azure Media Services.

**Network services** This includes Azure features such as Virtual Networks, ExpressRoute, Azure DNS, Azure Traffic Manager, and the Azure Content Delivery Network.

When migrating an application, it is worthwhile to have some understanding of the different services available in Azure because you might be able to use them to simplify the migration of your application and improve its robustness.

**The new world: Azure Resource Manager**

The Azure Resource Manager is the new methodology for deploying resources.

**What is it?**

Since it went into public preview, the Azure Service Management (ASM) deployment model has been used to deploy services. In the Azure portal, services managed with ASM are referred to as *classic*. In 2015, Microsoft introduced the Resource Manager deployment model as a modern, more functional replacement for ASM. The Resource Manager deployment model is recommended for all new Azure workloads.

These deployment models are often referred to as *control planes* because they are used to control services, not just to deploy them. This is different from a data plane, which manages the data used by a service. Typically, your running Azure infrastructure will contain many resources, but some of the resources will be related to one another in some way, such as all being the component services required to run a web application. For example, you might have two VMs running the web application, using a database to store data, and residing in the same virtual network. With Resource Manager, you deploy these assets into the same resource group and manage and monitor them together. You can deploy, update, or delete all of the resources in a resource group in one operation.

In this example, the resource group would contain the following:

* VM1
* VM2
* Virtual network
* Storage account
* Azure SQL Database

You can also create a template that precisely defines all the Resource Manager resources in a deployment. You can then deploy this Resource Manager template into a resource group as a single control-plane operation, with Resource Manager in Azure ensuring that resources are deployed correctly. After deployment, Resource Manager provides security, auditing, and tagging features to help you manage your resources.

**Why use Resource Manager?**

There are several advantages to using Resource Manager. The deployment is faster because resources can be deployed in parallel rather than sequentially as they are in ASM. The Resource Manager model enables each service to have its own service provider, and they can update it as needed independently of the other services. Azure Storage has its own service provider, VMs have their own service provider, and so on. With the ASM model, all services had to be updated at one time, so if one service was finished and the rest were not, the one that was ready had to wait on the others before it could be released. Here are some of the other major advantages to the Resource Manager model:

Deployment using templates

* You can create a reusable (JSON) template that can be used to deploy all of the resources for a specific solution in one fell swoop. You no longer have to create a VM in the portal, wait for it to finish, then create the next VM, and so on.
* You can use the template to redeploy the same resources repeatedly. For example, you may set up the resources in a test environment and find that it doesn’t fit your needs. You can delete the resource group, which removes all of the resources for you, then tweak your template and try again. If you only want to make changes to the resources deployed, you can just change the template and deploy it again, and Resource Manager will change the resources to conform to the new template.
* You can take that template and easily re-create multiple versions of your infrastructure, such as staging and production. You can parameterize fields such as the VM name, network name, storage account name, etc., and load the template repeatedly, using different parameters.
* Resource Manager can identify dependencies in a template but allows you to specify additional dependencies if necessary. For example, you wouldn’t want to deploy a virtual machine before creating the storage account for the VHD files that are used for the OS and data disks.

**Security**

* You can use the new Role-Based Access Control (RBAC) to control access to the resources in the group. For example, you can assign the Owner role to a user, giving that user full administrative privileges to those resources in the group but not to other resources in the subscription. Other roles include Reader (you can read anything except secrets) and Contributor (you can do most anything except add or revoke access).

**Billing**

* To help organize all of the resources in a subscription for billing purposes, you can assign tags to each resource and then retrieve all of the billing information for a specific tag.

For example, if one department owns a web application and several related components, you can assign the same tag to all of those resources. Then, you can retrieve the billing for that department by retrieving the billing for that tag.