AZURE FUNDAMENTALS

Core Cloud Services - Azure Compute Options

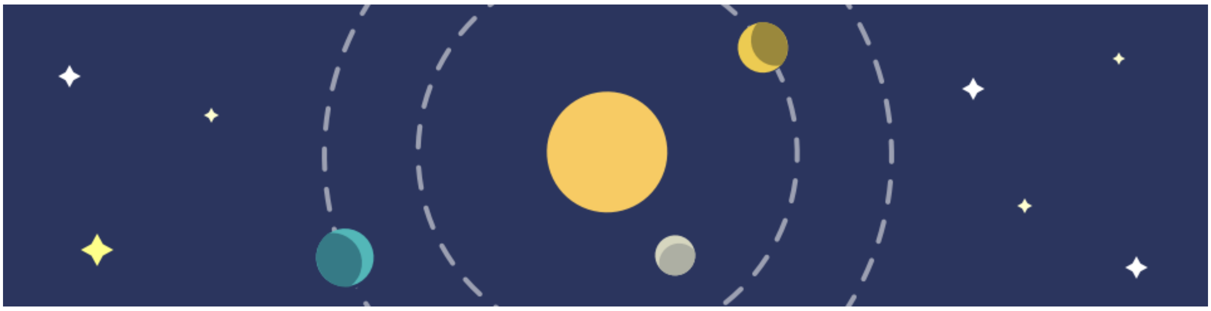
-Microsoft Learning

Disclaimer: The information contained in this document is completely owned and published by Microsoft Learning. I claim no credits to any of the content as I have just accumulated the available information for ease of read and learning. This document is/will/should not be used for any revenue systems.

Contents

Introduction

Imagine that you've landed your dream job as a developer at a new space technology firm. On your first day, you're assigned to help researchers analyze a large dataset being generated for a cutting-edge research project to explore water on Mars — and time is of the essence. But there's a problem; you don't have any free servers to do the work. And even if they did appear, you'd need to invest a lot of time to set them up and install software.



Of course you could ask to buy new equipment, but your department's budget is tight. Plus, you don't want to buy more hardware than needed, not only because you want to make a good impression with leadership, but also because you just don't know how much data will be generated by this project.

Ideally, you'd obtain the resources you need to do the work without too much administration — and simply configure them to do the work. And you'd pay only for the compute resources you need while you're using them.

This is exactly what we can do in Azure. We can create compute resources, configure them to do the work we need, and pay only for what we use.

Learning objectives

In this module, you will:

- Identify compute options in Azure
- Select compute options that are appropriate for your business

Essential Azure compute concepts

Your research team has collected massive amounts of image data that might lead to a discovery on Mars. They need to perform computationally intense data processing but don't have the equipment to do the work. Let's see why Azure is a good choice to do the data analysis.

What is Azure compute?

Azure compute is an on-demand computing service for running cloud-based applications. It provides computing resources like multi-core processors and supercomputers via virtual machines and containers. It also provides serverless computing to run apps without requiring infrastructure setup or configuration. The resources are available on-demand and can typically be created in minutes or even seconds. You pay only for the resources you use and only for as long as you're using them.

There are four common techniques for performing compute in Azure:

- Virtual machines
- Containers
- Azure App Service
- Serverless computing

What are virtual machines?

Virtual machines, or VMs, are software emulations of physical computers. They include a virtual processor, memory, storage, and networking resources. They host an operating system (OS), and you're able to install and run software just like a physical computer. And by using a remote desktop client, you can use and control the virtual machine as if you were sitting in front of it.

What are containers?

Containers are a virtualization environment for running applications. Just like virtual machines, containers are run on top of a host operating system but unlike VMs, they don't include an operating system for the apps running *inside* the container. Instead, containers bundle the libraries and components needed to run the application and use the existing host OS running the container. For example, if five containers are running on a server with a specific Linux kernel, all five containers and the apps within them share that same Linux kernel.

What is Azure App Service?

Azure App Service is a platform-as-a-service (PaaS) offering in Azure that is designed to host enterprise-grade web-oriented applications. You can meet rigorous performance, scalability, security, and compliance requirements while using a fully managed platform to perform infrastructure maintenance.

What is Serverless Computing?

Serverless computing is a cloud-hosted execution environment that runs your code but completely abstracts the underlying hosting environment. You create an instance of the service, and you add your code; no infrastructure configuration or maintenance is required, or even allowed.

Which computing strategy is right for me?

You don't need to take an "all or nothing" approach when choosing a cloud computing strategy. Virtual machines, containers, App Service, and serverless computing each provide benefits as well as tradeoffs against other options.

For example, although serverless computing removes the need for you to manage infrastructure, serverless computing expects work to be completed quickly; usually within seconds or less. Therefore, you might run your core application on a virtual machine or container but offload some of the data processing onto a serverless app.

Let's look at each option more closely to help you decide when to use each service.

Explore Azure Virtual Machines

Azure Virtual Machines (VMs) let you create and use virtual machines in the cloud. They provide infrastructure as a service (IaaS) in the form of a virtualized server and can be used in many ways. Just like a physical computer, you can customize all of the software running on the VM. VMs are an ideal choice when you need:

- Total control over the operating system (OS)
- The ability to run custom software, or

- To use custom hosting configurations

You can create and provision a VM in minutes when you select a pre-configured VM image. Selecting an image is one of the most important decisions you'll make when creating a VM. An image is a template used to create a VM. These templates already include an OS and often other software, like development tools or web hosting environments.

Moving to the cloud with VMs

VMs are also an excellent choice when moving from a physical server to the cloud ("lift and shift"). You can create an image of the physical server and host it within a VM with little or no changes. Just like a physical on-premises server, you must maintain the VM. Update the OS and the software it runs.

Scaling VMs in Azure

You can run single VMs for testing, development, or minor tasks, or group VMs together to provide high availability, scalability, and redundancy. Azure has several features so that no matter what your uptime requirements are, Azure can meet them. These features include:

- Availability sets
- Virtual Machine Scale Sets
- Azure Batch

What are availability sets?

An **availability set** is a logical grouping of two or more VMs that help keep your application available during planned or unplanned maintenance.

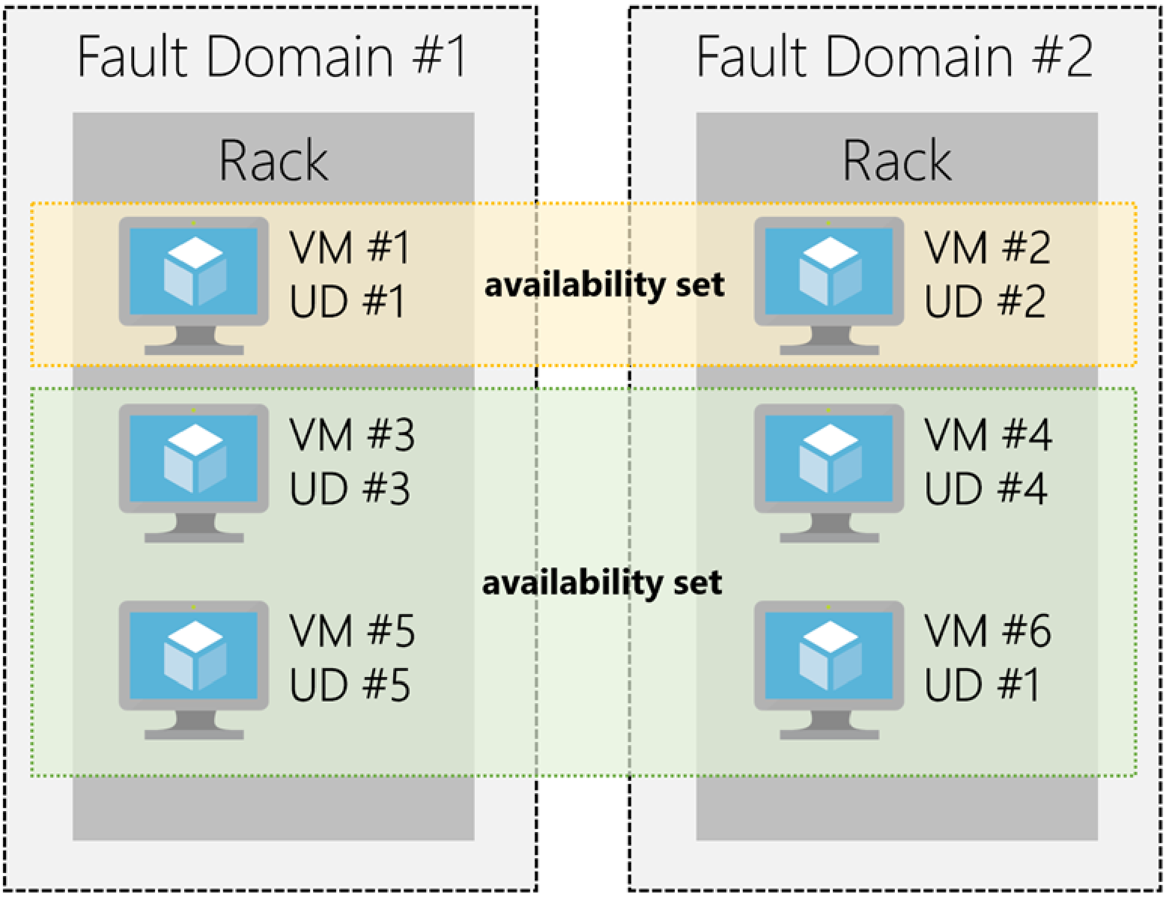
A *planned maintenance event* is when the underlying Azure fabric that hosts VMs is updated by Microsoft. A planned maintenance event is done to patch security vulnerabilities, improve performance, and add or update features. Most of the time these updates are done without any impact to the guest VMs. But sometimes VMs require a reboot to complete an update. When the VM is part of an availability set, the Azure fabric updates are sequenced so not all of the associated VMs are rebooted at the same time. VMs are put into different *update domains*. Update domains indicate groups of VMs and underlying physical hardware that can be rebooted at the same time. Update domains are a logical part of each data center and are implemented with software and logic.

Unplanned maintenance events involve a hardware failure in the data center, such as a power outage or disk failure. VMs that are part of an availability set automatically switch to a working physical server so the VM continues to run. The group of virtual machines that share common hardware are in the same *fault domain*. A fault domain is essentially a rack of servers. It provides the physical separation of your workload across different power, cooling, and network hardware that support the physical servers in the data center server racks. In the event the hardware that supports a server rack becomes unavailable, only that rack of servers is affected by the outage.

With an availability set, you get:

- Up to three fault domains that each have a server rack with dedicated power and network resources
- Five logical update domains

Your VMs are then sequentially placed across the fault and update domains. The following diagram shows an example where you have six VMs in an availability set distributed across the two fault domains and five update domains.



There's no cost for an availability set. You only pay for the VMs within the availability set. We highly recommend that you place each workload in an availability set to avoid a single point of failure in your VM architecture.

What are virtual machine scale sets?

Azure Virtual Machine Scale Sets let you create and manage a group of identical, load balanced VMs. Imagine you're running a website that enables scientists to upload astronomy images that need to be processed. If you duplicated the VM, you'd normally need to configure an additional service to route requests between multiple instances of the website. VM Scale Sets could do that work for you.

Scale sets allow you to centrally manage, configure, and update a large number of VMs in minutes to provide highly available applications. The number of VM instances can automatically increase or decrease in response to demand or a defined schedule. With VM Scale Sets, you can build large-scale services for areas such as compute, big data, and container workloads.

What is Azure Batch?

Azure Batch enables large-scale job scheduling and compute management with the ability to scale to tens, hundreds, or thousands of VMs.

When you're ready to run a job, Batch:

- Starts a pool of compute VMs for you
- Installs applications and staging data
- Runs jobs with as many tasks as you have
- Identifies failures
- Requeues work
- Scales down the pool as work completes

There may be situations in which you need raw computing power or supercomputer level compute power. Azure provides these capabilities.

Explore Containers in Azure

If you wish to run multiple instances of an application on a single virtual machine, containers are an excellent choice. The container orchestrator can start, stop, and scale out application instances as needed.

Containers are meant to be lightweight, created, scaled out, and stopped dynamically. This design allows you to respond quickly to changes in demand or failure.

Another benefit of containers is you can run multiple isolated applications on a single VM host. Since containers are secured and isolated, you don't need separate VMs for each app.

Containers in Azure

Azure supports Docker containers, and there are several ways to manage containers in Azure.

- Azure Container Instances (ACI)
- Azure Kubernetes Service (AKS)

Azure Container Instances

Azure Container Instances (ACI) offers the fastest and simplest way to run a container in Azure. You don't have to manage any virtual machines or configure any additional services. It is a PaaS offering that allows you to upload your containers and execute them directly.

Azure Kubernetes Service

The task of automating and managing and interacting with a large number of containers is known as orchestration. Azure Kubernetes Service (AKS) is a complete orchestration service for containers with distributed architectures with multiple containers.

Using containers in your solutions

Containers are often used to create solutions using a *microservice architecture*. This is where you break solutions into smaller, independent pieces. For example, you may split a website into a container hosting your front end, another hosting your back end, and a third for storage. This allows you to separate portions of your app into logical sections that can be maintained, scaled, or updated independently.

What is a microservice?

Imagine your website backend has reached capacity but the front end and storage aren't being stressed. You could scale the back end separately to improve performance, or you could decide to use a different storage service. Or you could even replace the storage container without affecting the rest of the application.

Explore Azure App Service

Azure App Service enables you to build and host web apps, background jobs, mobile backends, and RESTful APIs in the programming language of your choice without managing infrastructure. It offers auto-scaling and high availability, supports both Windows and Linux, and enables automated deployments from GitHub, Azure DevOps, or any Git repo to support a continuous deployment model.

This platform as a service (PaaS) allows you to focus on the website and API logic while Azure takes care of the infrastructure to run and scale your web applications.

App Service costs

You pay for the Azure compute resources your app uses while it processes requests based on the App Service Plan you choose. The App Service plan determines how much hardware is devoted to your host - for example, whether it's dedicated or shared hardware, and how much memory is reserved for it. There is even a *free* tier you can use to host small, low-traffic sites.

Types of web apps

With Azure App Service, you can host most common web app styles including:

- Web Apps
- API Apps
- WebJobs
- Mobile Apps

Azure App Service handles most of the infrastructure decisions you deal with in hosting web apps: deployment and management are integrated into the platform, endpoints can be secured, sites can be scaled quickly to handle high traffic loads, and the built-in load balancing and traffic manager provide high availability. All of these app styles are hosted in the same infrastructure and share these benefits. This makes App Service the ideal choice to host web-oriented applications.

Web Apps

App Service includes full support for hosting web apps using ASP.NET, ASP.NET Core, Java, Ruby, Node.js, PHP, or Python. You can choose either Windows or Linux as the host operating system.

API Apps

Much like hosting a website, you can build REST-based Web APIs using your choice of language and framework. You get full Swagger support, and the ability to package and publish your API in the Azure Marketplace. The produced apps can be consumed from any HTTP(s) based client.

Web Jobs

WebJobs allows you to run a program (.exe, Java, PHP, Python or Node.js) or script (.cmd, .bat, PowerShell, or Bash) in the same context as a web app, API app, or mobile app. They can be scheduled, or run by a trigger. This is often used to run background tasks as part of your application logic.

Mobile Apps

Use the Mobile Apps feature of Azure App Service to quickly build a back-end for iOS and Android apps. With just a few clicks in the Azure portal you can:

- Store mobile app data in a cloud-based SQL database
- Authenticate customers against common social providers such as MSA, Google, Twitter and Facebook
- Send push notifications
- Execute custom back-end logic in C# or Node.js

On the mobile app side, there is SDK support for native iOS & Android, Xamarin, and React native apps.

Explore Serverless computing in Azure

With *serverless* computing, Azure takes care of managing the server infrastructure and allocation/deallocation of resources based on demand. Infrastructure isn't your responsibility. Scaling and performance are handled automatically, and you are billed only for the exact resources you use. There's no need to even reserve capacity.

You focus solely on the logic you need to execute and the *trigger* that is used to run your code. You configure your serverless apps to respond to *events*. This could be a REST endpoint, a periodic timer, or even a message received from another Azure service. The serverless app runs only when it's triggered by an event.

Serverless computing in Azure

Azure has two implementations of serverless compute:

- **Azure Functions** which can execute code in almost any modern language.
- **Azure Logic Apps** which are designed in a web-based designer and can execute logic triggered by Azure services without writing any code.

Azure Functions

When you're concerned only about the code running your service, and not the underlying platform or infrastructure, Azure Functions are ideal. They're commonly used when you need to perform work in response to an event, often via a REST request, timer, or message from another Azure service and when that work can be completed quickly, within seconds or less.

Azure Functions scale automatically based on demand, so they're a solid choice when demand is variable. For example, you may be receiving messages from an IoT solution used to monitor a fleet of delivery vehicles. You'll likely have more data arriving during business hours.

Using a VM-based approach, you'd incur costs even when the VM is idle. With functions, Azure runs your code when it's triggered and automatically deallocates resources when the function is finished. In this model, you're only charged for the CPU time used while your function runs.

Furthermore, Azure Functions can be either stateless (the default) where they behave as if they're restarted every time they respond to an event), or stateful (called "Durable Functions") where a context is passed through the function to track prior activity.

Azure Logic Apps

Azure Logic Apps are similar to Functions - both enable you to trigger logic based on an event. Where Functions execute code, Logic Apps execute *workflows* built from predefined logic blocks. They are specifically designed to automate your business processes.

You create Logic App workflows using a visual designer on the Azure Portal or in Visual Studio. The workflows are persisted as a JSON file with a known workflow schema.

Azure provides over 200 different connectors and processing blocks to interact with different services - including most popular enterprise apps. You can also build custom connectors and workflow steps if the service you need to interact with isn't covered. You then use the visual designer to link connectors and blocks together, passing data through the workflow to do custom processing - often all without writing any code.

As an example, let's say a ticket arrives in ZenDesk. You could:

1. Detect the intent of the message with cognitive services
2. Create an item in Sharepoint to track the issue
3. If the customer isn't in your database, add them to your Dynamics 365 CRM system
4. Send a follow-up email to acknowledge their request

All of that could be designed in a visual designer making it easy to see the logic flow which is ideal for a business analyst role.

Functions vs. Logic Apps

Functions and Logic Apps can both create complex orchestrations. An orchestration is a collection of functions or steps, that are executed to accomplish a complex task. With Azure Functions, you write code to complete each step, with Logic Apps, you use a GUI to define the actions and how they relate to one another.

You can mix and match services when you build an orchestration, calling functions from logic apps and calling logic apps from functions. Here are some common differences between the two.

-	**Functions**	**Logic Apps**
State	Normally stateless, but Durable Functions provide state	Stateful
Development	Code-first (imperative)	Designer-first (declarative)
Connectivity	About a dozen built-in binding types, write code for custom bindings	Large collection of connectors, Enterprise Integration Pack for B2B scenarios, build custom connectors
Actions	Each activity is an Azure function; write code for activity functions	Large collection of ready-made actions
Monitoring	Azure Application Insights	Azure portal, Log Analytics
Management	REST API, Visual Studio	Azure portal, REST API, PowerShell, Visual Studio
Execution context	Can run locally or in the cloud	Runs only in the cloud.

Summary

Azure provides multiple services to perform cloud compute, but choosing the right service depends on your business needs. Remember that there are some overlaps in capabilities. For example, you could use either Azure containers or Azure Functions as part of a serverless architecture. But ultimately, making the right decision depends on both the service capability and the abilities of your development team.