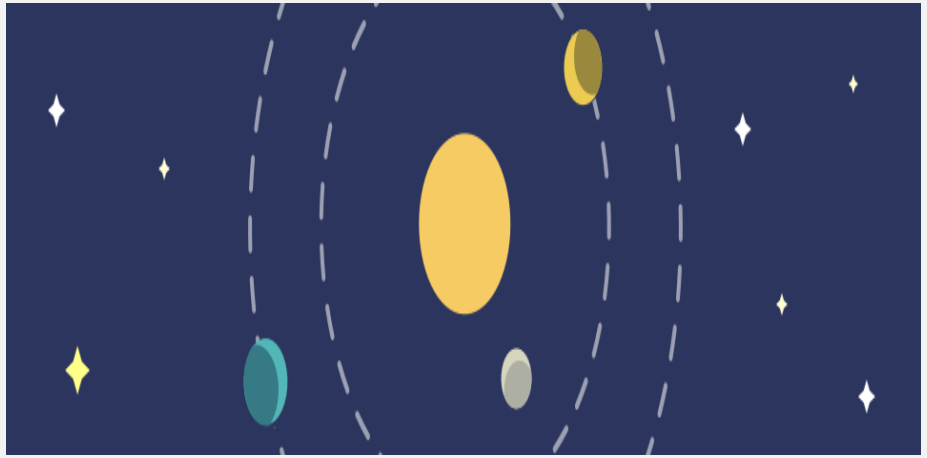
Real World Example

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.

Azure Compute to Rescue ??

What is Azure compute?

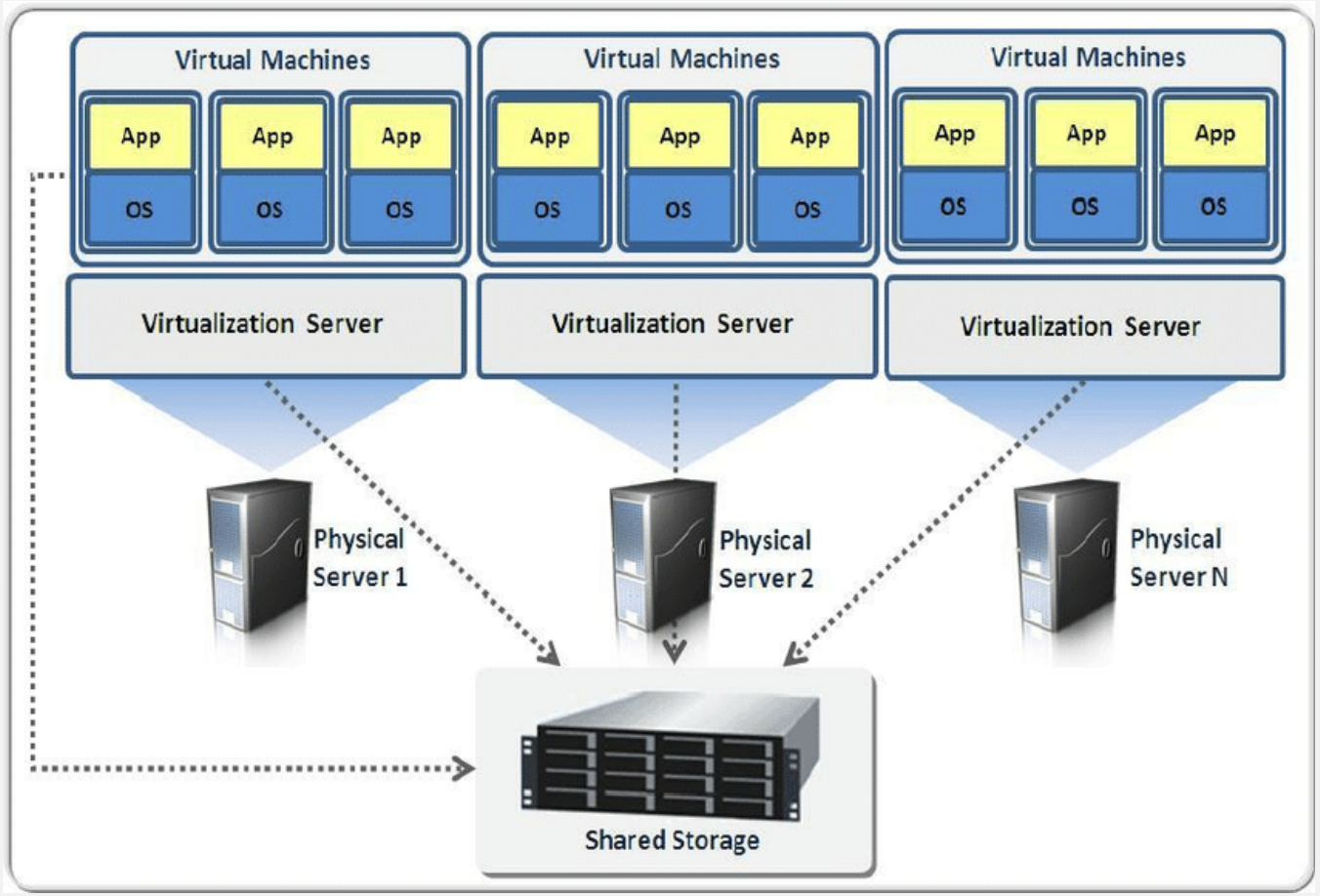
Azure compute is an on-demand computing service for running cloud-based applications.
It provides computing resources like multi-core processors and supercomputers via virtual machines and containers.
It also provides server less computing to run apps without requiring infrastructure setup or configuration.
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.



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
- To use custom hosting configurations

An Azure VM gives you the flexibility of virtualization without having to buy and maintain the physical hardware that runs the VM. However, you still need to maintain the VM—that is, configure, update, and maintain the software that runs on the VM.

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.

HW: List out all the Windows Images available in the portal

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, containers 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.

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.

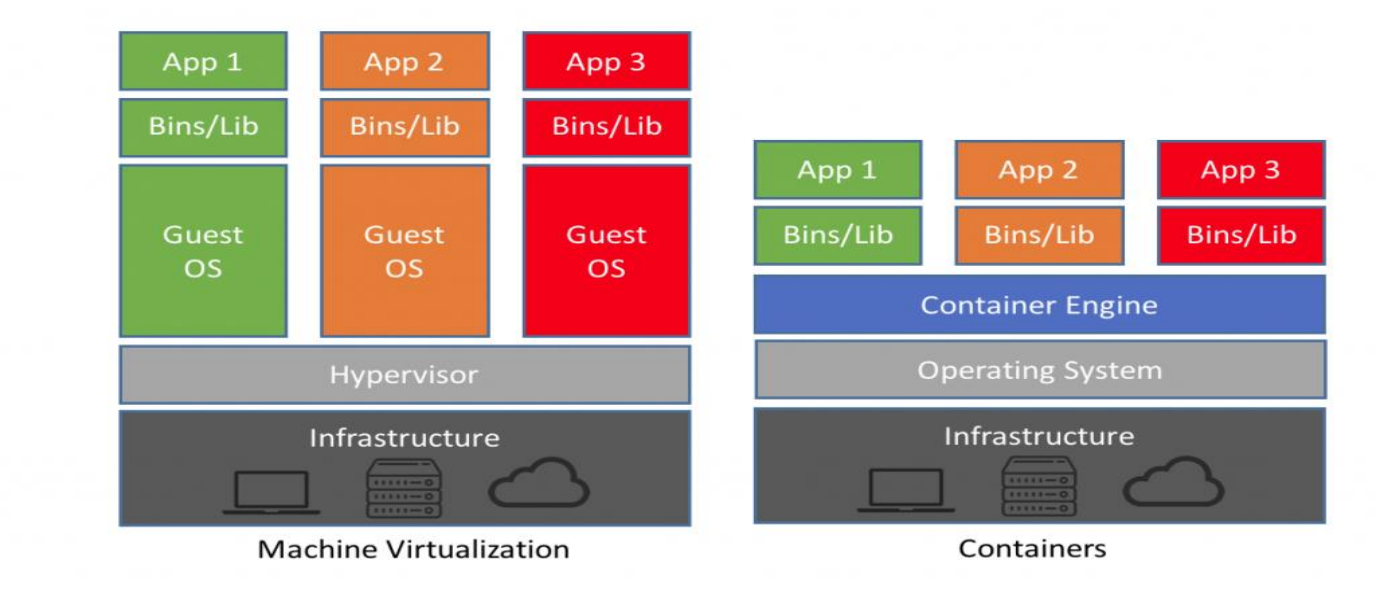
What is Serverless Computing?
Serverless computing is a cloud-hosted execution environment that runs your code but completely abstracts the underlying hosting environment

Containers in Azure

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

- A container is a modified runtime environment built on top of a host OS that executes your application.
- A container doesn't use virtualization layer of VM , so it doesn't waste resources simulating virtual hardware with a redundant OS.

VMs versus containers



HANDS ON

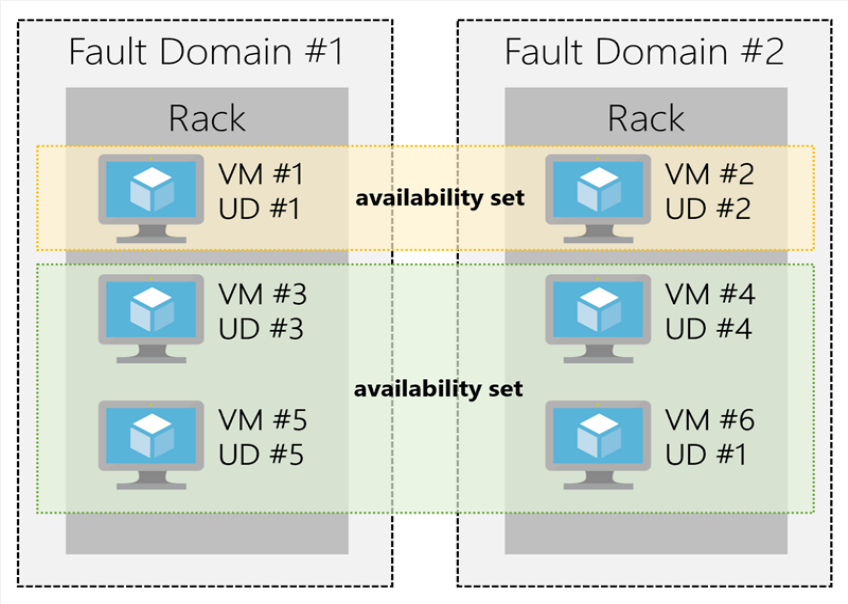
- Create a Virtual Machine on already configured Vnet
- Try to RDP into it
- Attach Public IP to it
- Try RDP
- Setup and auto shutdown Schedule

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. 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.

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.



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 which then can be increased to a maximum of 20

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. Virtual Machine Scale Sets could do that work for you.

- VM can be automatically scaled vertically and horizontally using automatic and manual methods.
- In Horizontal scaling the azure monitor monitors, and adds or removes vm .
- Vertical scaling keeps the number of VM same but make the vm more or less powerful.
- In PAAS there is no need to have auto scaling.

Configuring Auto scaling

- Automatic scaling provides right number of VM to handle load of the application.
- It enables handling the load by adding VM when required and scaling down when the load is low to ensure cost is optimized.
- Having a Minimum ensures that the even when there is no load particular number of VM are running and maximum would mean the maximum extent to which VM can scale.

VM types and Sizes

HW – List Down VM types available in Azure

- Disk throughput is measured in IOPS and MBps
- Data disk can operate in cached mode which is read write /read only or None.
- Expected Network Bandwidth is the maximum aggregated network throughput allocated to each VM across all NIC across all destinations. Depends on max nics which can be allocated and the aggregated network throughput of the machine.

General purpose VM

- General purpose VM provide balanced CPU to memory ratios.
- Ideal for testing and development small to medium databases.
- A Series and AV2 series - are basic VM
- D - Series - Designed to run applications which require higher compute power and temporary disk performance.
- DV2 - A follow-up to the D series machines CPU is 35% faster than the D series CPU.Memory and disk configuration are similar as d series.
- DV3 - It features the same processors as the DV2 series but in hyper threaded configuration

Memory Optimized Virtual machines

- These Virtual machines offer high memory to CPU ration and are great for relational database
- M series provides the highest vCPU count up to 128 cpu and largest memory 3.8 TB,ideal for extremely large Databases
- Dv2 series, D Series, G-Series and the DS/GS counterparts ideal for applications better temporary storage performance and high vCPU demand.
- Ev3 series - provides better value proposition for most general purpose workloads.

Storage Optimized

- Storage optimized workloads offer high disk throughput and IO and ideal for Big data no SQL databases
- LS series offers 32 vCPU

GPU optimized

- Specialized VM which are available with a single or multiple NVIDIA GPU designed for graphic intensive work.
- NC,NV are optimized for the same

GPU optimized

- Specialized VM which are available with a single or multiple NVIDIA GPU designed for graphic intensive work.
- NC,NV are optimized for the same

Containers in Azure

Azure supports Docker containers (a standardized container model), 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 with automatic elastic scale.

Azure Kubernetes Service

The task of automating, 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.

Azure App Service

- Azure App Service enables you to build and host web apps, background jobs, mobile backend, and RESTful APIs in the programming language of your choice without managing infrastructure.
- It offers automatic scaling and high availability. App Service 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 handles 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.

Types of web apps

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

- Web Apps
- API Apps
- WebJobs
- Mobile Apps

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
Web Jobs 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. WebJobs are often used to run background tasks as part of your application logic.

Mobile app back-ends
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

Serverless computing in Azure

Serverless computing is the abstraction of servers, infrastructure, and OSs. 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.

Serverless computing encompasses three ideas: the abstraction of servers, an event-driven scale, and micro-billing:

Abstraction of servers: Serverless computing abstracts the servers you run on. You never explicitly reserve server instances; the platform manages that for you. Each function execution can run on a different compute instance, and this execution context is transparent to the code. With serverless architecture, you simply deploy your code, which then runs with high availability.

Event-driven scale: Serverless computing is an excellent fit for workloads that respond to incoming events. Events include triggers by timers (for example, if a function needs to run every day at 10:00 AM UTC), HTTP (API and webhook scenarios), queues (for example, with order processing), and much more. Instead of writing an entire application, the developer authors a function, which contains both code and metadata about its triggers and bindings. The platform automatically schedules the function to run and scales the number of compute instances based on the rate of incoming events. Triggers define how a function is invoked and bindings provide a declarative way to connect to services from within the code.

Micro-billing: Traditional computing has the notion of per-second billing, but often, that's not as useful as it seems. Even if a customer's website gets only one hit a day, they still pay for a full day's worth of availability. With serverless computing, they pay only for the time their code runs.

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.

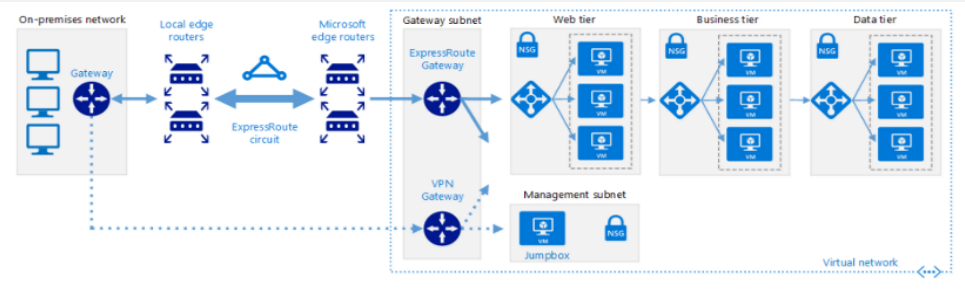
Azure Functions

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 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* designed to automate business scenarios and built from predefined logic blocks. Every logic app workflow starts with a trigger, which fires when a specific event happens or when newly available data meets specific criteria. Many triggers include basic scheduling capabilities, so developers can specify how regularly their workloads will run.

Homework



1. Create a 2 Vnet create – 2 Vnets with 2 Subnets (https://docs.microsoft.com/en-us/azure/virtual-network/quick-create-portal)
2. Establish peering between two Vnets (https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-peering-overview)
3. Create 2 VM in 2 Vnets with Public IP (https://docs.microsoft.com/en-us/azure/virtual-machines/windows/quick-create-portal)
4. Ping VM1 from Vm2 and vice versa.