

Microsoft Azure Solutions Architect Expert (AZ-305) Master Cheat Sheet

1. Azure basics.

Azure basics

Interacting with Azure

- Azure is based on REST APIs
- You can use Portal, PowerShell, Azure CLI that wrap REST APIs.
- **Azure Cloud Shell**: Browser-accessible shell on portal that can run PowerShell, Azure CLI and even more like git/bash/pip/maven etc.
- 💡 Azure CLI can often handle everything that other wrappers can and even more.

Service-level agreement (SLA)

- A guarantee that Azure gives to customers for different offerings.
- Guarantees Monthly Uptime Percentage
- $\text{Monthly Uptime \%} = (\text{Maximum Available Minutes} - \text{Downtime}) / \text{Maximum Available Minutes} \times 100$

Regions

- Azure has different regions
- Each Azure Region has one or more (often 3) *availability zones*.
 - Each availability zone is made up one or more *data-centers*.
 - Data centers have independent power, cooling and networking.
- Each region includes a *pair* in its country (>500 kms away if it's possible)
 - Pairs enables *system update isolation* where regions are updated in queue[^fn1]
 - Azure region pair is highly prioritized during recovery
 - Services with geo-redundant storage uses paired region automatically.

Resource Group

- Logical group to manage *resources* together
- Groups values e.g. analyzing and forecasting resource consumption and spending.
- You can create policies on resource group
- A **resource** is an object in Azure (Azure object)

Azure Resource Manager (ARM)

- Each object in Azure has ARM files associated with it.
- Can be deployed directly from Visual Studio
- They are JSON text files.
 - *\$schema (required)* : URL of the JSON schema file describing the version.
 - *contentVersion (required)* : Version of the template (e.g. 1.0.0.0)
 - *resources (required)*: Resource types that's deployed or updated in the group.
 - *parameters*: Customizable values that are provided when deployment executed.
 - *variables*: JSON fragments in template to simplify language expressions.
 - *outputs*: Values that are returned after deployment.
- **!** Secure any username, password parameters in JSON files.
- Usually parameters (`azuredeploy.parameters.json`) and the file (`azuredeploy.json`) is separated.
- **!** Hard to create from scratch.
 - **!** Have a base and modify later.
 - Create a resource, copy its auto-generated ARM from *Automation blade*.
 - Use **Azure QuickStart templates**
 - Maintained by Microsoft + Community (on GitHub)
 - Provides "Deploy to Azure" button
- You can control how things are deployed using **Azure Policy** on resource group, subscription, or management group level.

Egress charges

- Moving data to Azure mostly is free.
- From Azure to outside (without ExpressRoute or Content Delivery Network) you get extra egress charges.

2.1. Monitoring

Monitoring

- Act of collecting & analyzing data to determine the performance, health, and availability of applications and their depended resources.
- 🚨 Understanding the operation of applications and alerts helps fixing problems before they occur.

Azure monitoring services

- **Application Monitoring:** • Application Insights
- **Deep Infrastructure Monitoring:** • Log Analytics • Management Solutions • Network Monitoring • Service Map
- **Core Monitoring:** • Azure Monitor • Advisor • Service Health • Activity Log
- **Shared Capabilities:** • Alerts • Dashboards • Metrics Explorer

Azure monitoring costs

- Use [Pricing Calculator](#) for a specific resource.
- Review estimated costs when creating a resource
- See spent costs through Subscription blade
 - You can task resources Cost analysis with e.g. *costCenter: marketing* to filter in cost analysis view.
 - 🚨 Recommended to tag & export.
 - You can see your bill in "Invoices", opt-in for PDF invoices & download (more options available in *Azure Account Center*)
 - External services billed separately
- Get notifications through Azure Advisor or Azure Cost Management for anomalies and overspending risks.

Azure Monitor

- PaaS (Platform as a Service)
- Pipeline for metric log data coming from any Azure resource provider.
- Fastest telemetry pipeline: Faster than Log Analytics.
- Collected data is saved in Log Analytics to analyze, monitor, visualize metrics and query and analyze logs.
 - *Visualize:* • Dashboard • Views • Power BI • Workbooks
 - *Analyze:* • Metrics Explorer • Log Analytics

- *Respond:* • Alerts • Autoscale
- *Integrate:* • Event Hubs • Logic Apps • Ingest & Export APIs
- You can use alerts to proactively notify you based on rules with Azure Alerts.
- You can access through
 - Azure Insight & Analytics (OMS portal that's now legacy)
 - On Azure portal there's an existing resource called Azure Monitor.
 - Rest API, PowerShell & CLI

Data types

- Two levels of logs: resource (*metrics*) and platform (*activity log, diagnostic log*).
- All data types can be queried in Portal, PowerShell, Rest API or CLI.

Metrics

- On resource level
- All numerical values including
 - All Application Insights data / telemetry
 - System health from VMs (uses HyperV metrics without any configuration)
- Can be visualized & queried
- 30 days free to store & query
- E.g. requests and errors, average response time, infrastructure metrics

Activity Log

- On platform level
- "Outside operations" that are subscription-level events
- Previously known as "Audit Logs" or "Operational Logs"
- Determine "what, who and when" for any write operation taken on the resources.
- 90 days free to store, for more send to Log Analytics (["connect" button in Activity Logs blade of Log Analytic](#)), archive to storage account ([click on Export to Event Hubs in Azure Monitor](#)) or stream to other services ([click on Export to Event Hubs in Azure Monitor](#)).
 - During export you create a Log Profile you control how/where/what is exported for how long.

Activity log categories

- **Administrative**
 - Every API call to Azure Resource Manager

- E.g. create virtual machine or delete network security group
- **Service health**
 - Any service health incidents occurred in Azure
 - Come in 5 varieties: • Action Required • Assisted Recovery • Incident • Maintenance • Information • Security.
 - E.g. SQL Azure in East US is experiencing downtime.
- **Resource health**
 - Status of resources: • Available • Unavailable • Degraded • Unknown
 - E.g. Virtual Machine health status changed to unavailable
- **Alert**
 - All activations of Azure alerts.
 - E.g. CPU % on a VM has been over 80 for the past 5 minutes
- **Autoscale**
 - Events in autoscale engine defined by your settings.
 - E.g. Scale up action failed
- **Recommendation**
 - How to better utilize your resources.
- **Security**
 - Events by Azure Security Center
 - E.g. Suspicious double extension file executed.
- **Policy**
 - All effect action operations performed by Azure Policy.
 - Types include audit and deny

Diagnostic Log

- On platform level
- "Inside operations" within the resource
- Resource-specific with common scheme where resources include their own properties.
- E.g. IIS logs, web server logging, failed request tracking
- Can be streamed (to Power BI, Azure Functions etc.), added in blob storage, and sent directly to Log Analytics

Azure Alerts

- All alert creation for metrics, logs and activity log across Azure Monitor, Log Analytics, and Application Insights
- Separation of operational and configuration views:
 - **Alert Rules:** Definition of the condition that triggers an alert

- **Fired Alerts:** An instance of the alert rule firing
- Flow of alerts
 - Set-up alert rule
 - Target resource (e.g. storage account)
 - **Signal:**
 - Types are Metric, Activity log, Application Insights and Log
 - You can have multi-dimensional metrics & monitor multiple metrics with a single rule (currently up to two)
 - **Criteria**
 - Logic test: e.g. six-hour period when capacity is over 10 MB
 - Action group (= actions to do)
 - Grouping of different actions to take when the alert is triggered
 - Each action has name & action type e.g. email/sms/push/voice/webhook/automation runbook
 - **! Applied rate limiting:**
 - SMS: No more than 1 SMS every 5 minutes.
 - Voice: No more than 1 Voice call every 5 minutes.
 - Email: No more than 100 emails in an hour.
 - Other actions are not rate limited.
 - Set alert rule name, description, severity (can be informational, warning, error, critical)
 - Log alerts
 - Defined by Log Query (by Log Analytics), Time period, frequency, and threshold.
 - *Number of results alert rules* always creates a single alert, while *Metric measurement alert rule* creates an alert for each object that exceeds the threshold

Azure Advisor

- Uses telemetry & application configurations to give personalized recommendations and guidance for
 - high availability, security, performance, cost effectiveness (*monitors unused resources and spent*).
- Common resource, free for all users
- You can download, filter, postpone, and dismiss recommendations.
- You can customize by excluding subscription/resource groups, configuring utilization rules (e.g. you can as a subscription owner set CPU to lower threshold)

Log Analytics

- PaaS (Platform as a Service)
- It's also referred as (newer names)
 - **Azure Monitor Logs**
 - Azure Monitor log data is stored in Log Analytics
 - The term is changed from "Log Analytics" to "Azure Monitor Logs" ([see](#))
 - **Log Search**
 - "Log Search" interface in Azure Monitor (Monitor -> Logs)
 - Or Log Analytics Workspace -> Logs
- Two main features
 - Log Analytics Workspace where Azure Monitor stores its data
 - Log search feature; collect, correlate, search, and act on data with any schema.
- Business value:
 - *Assessing updates*: From logs can guess average patching time
 - *Change tracking*: Abnormal behavior from a specific account by tracking changes throughout the environment
- Free to store logs for 90 days of charge
 - You can export to Excel, PowerBI, or use API to send data or get.

Sending data to Log Analytics

- Log analytics uses push-model i.e. data is pushed to it.
- Data can be pushed from integrated sources including
 - Connected Azure sources e.g. IIS logs, custom text logs with custom fields, error level etc.
 - Office 365, Azure Automation, Back-ups
 - !⚠ You cannot change schema on ingestion time for integrated resources, only on query time.
- Data can also be pushed Linux & Windows systems with an agent
 - Agents include
 - Log analytics agent
 - Operations Manager
 - Allows using existing investments with SCOM (System Center Operations Manager)
 - SCOM agents communicate with SCOM Server over TLS 1.2 which forward events and performance data to Log Analytics.
 - You can install agents using a script via Azure Automation DSC (desired state configuration)
 - Agents are already installed in cloud Windows VMs
- You can also use Log Analytics REST APIs to send custom data with any schema you want

Collect data across subscriptions

- Log analytics applies if in same subscription, or in same Azure Active Directory tenant.
- So single Log Analytics workspace can monitor across under same tenant.
- To collect data across subscriptions and tenants:
 - In customer subscription
 - Activity log (*export button*) => Event Hub
 - In Service provider subscription
 - Logic App (*When events are available in Event Hub -> Parse JSON (Body) -> Compose (Select Body in inputs) -> Send Data (Azure Log Analytics Data Collector)*)=> Log Analytics
- Azure Monitor Logs support query across multiple Log Analytics workspaces across subscriptions
 - E.g. `app("/subscriptions/b459b4f6-912x-46d5-9cb1-b43069212ab4/resourcegroups/Fabrikam/providers/microsoft.insights/components/fabrikamapp").requests | count`
 - Read more on [Microsoft Docs](#)

Querying Log Analytics

- Each data source has documentation (description & name) of its properties.
- 📌 Main query tables: *Heartbeat, Perf, Usage, Event, Syslog, Alert*.
- You can connect to Activity Logs in Activity Logs section and query with AzureActivity

Kusto Query Language (KQL)

- A pipe-through language to query log analytics data
- E.g.: `Event | where (EventLevelName == "Error") | where (TimeGenerated > ago(1days)) | summarize ErrorCount = count() by Computer | top 10 by ErrorCount desc`
- Can generate charts | `render timechart` that can be pinned to dashboard.

2.2. Azure Storage

Azure Storage

- PaaS (Platform as a Service)
- Store files, messages, tables, and other types of information.
 - Subservices: Blob, File, Queue, and Table.

- It's also used by IaaS virtual machines, and other PaaS cloud services.
- Three main roles:
 - Storage for Virtual Machines
 - **Disks** are persistent block storage for Azure IaaS virtual machines.
 - Azure disks handles creation of Storage account for you.
 - **Files** are fully managed file shares in the cloud.
 - Unstructured Data
 - **Blobs** are highly scalable, REST based cloud object store.
 - **Data Lake Store** is Hadoop Distributed File System (HDFS) as a service.
 - **Tables** are a key/value, auto-scaling NoSQL store.
 - **Cosmos DB** is a globally distributed database service
 - Structured Data
 - **Azure SQL DB** is fully managed database-as-a-service built on SQL
- A **snapshot** is a read-only version of a storage that's taken at a point in time.
 - Can be read, copied or deleted but not modified.
 - Provide a way to back up a blob as it appears at a moment in time

Azure Storage Explorer

- Access multiple accounts and subscriptions.
- Can manage Blob, Queue, Table, File, Cosmos DB and Data Lake storage.
 - Obtain shared access signature (SAS) keys.
- Authorization: SAS or Azure account.

Storage types

Queue

- More basic than ServiceBus without overhead
- 💡 Good for >80GB

Blobs

- Blob permissions
 - **Private**: Authentication (RBAC or storage key) to access
 - **Blob**: Anonymous access to blobs
 - **Container**: Anonymous access to blobs + containers
- Blob types

- **Block blob** is aimed at streaming and storing of objects such as media files and documents. Sequentially accessed.
- **Page blob** is optimized for random reads and writes. E.g. SQL & VMs uses it.
- **Append blob** is same as a **block blob** except data can only be added to the end of the blob. Blocks elsewhere cannot be deleted nor modified
- Blob storage access tiers
 - **Hot**: More frequently accessed
 - **Cool**: Less frequently accessed
 - **Archive**: Minimum frequency. Only in v2

Azure Storage Accounts

- Around 99.9% SLA
- Kinds
 - Legacy => **General purpose v1, Blob storage**
 - ! Can easily be migrated to v2
 - **General purpose v2**
 - Lowest price, pay per GB

General-purpose storage account

- Tables, queues, files, blobs and Azure virtual machine disks under a single account
- Performance tiers:
 - **Standard**: tables, queues, files, blobs, and Azure virtual machine disks
 - Magnetic drives (HDD) and provide the lowest cost per GB
 - Best for apps with bulk storage or where data is accessed infrequently.
 - **Premium**: Supports only Azure virtual machine disks
 - SSD, low latency performance
 - Best for I/O-intensive applications, like databases
 - ! Not possible to convert after deployment

Endpoints

- Default domain name e.g. `http://mystorageaccount.**blob**.core.windows.net`
- Two ways to configure
 - Direct CNAME mapping
 - Create DNS record (*NS record as TXT or MX*) for Azure URL
 - ! For subdomains you use A record.
 - Causes minor downtime as the domain is updated.

- Intermediary mapping with *asverify*
 - No downtime
 - First create CNAME (in 3rd party) e.g. **azverify**.docs.amk.com
 - Put it in Azure and it gets verified
 - Then create CNAME (in 3rd party) docs.amk.com
 - You can now delete azverify CNAME record.

Creating accounts

- You can use VNet to restrict the storage to only certain IP address subnets or VNets.
- ! Location must match the location of the resource group

Data Replication

- Replication strategies
 - **LRS – Locally Redundant Storage**
 - Copies: 1 copy in within a single facility in same region
 - **ZRS – Zone Redundant Storage**
 - Copies: 3 copies in 3 different availability zones in same region
 - 🗄️ Replicates synchronously
 - **Geo-redundant Storage (GRS)**
 - Copies: In Another data center in a secondary (paired) region
 - 🗄️ Replicates asynchronously.
 - ! Not available in all regions
 - **Read access geo redundant storage (RA-GRS)**
 - Copies: As in GRS
 - Read access to the replicate.
 - Blob service is myaccount.blob.core.windows.net, then your secondary endpoint is myaccount-secondary.blob.core.windows.net.
 - The access keys for your storage account are the same for both the primary and secondary endpoints.
- ? You can change your replication strategy later on.
 - From LRS: Free
 - GRS to RA-GRS: incur egress bandwidth charge (one time cost initially)
- ! If you create availability sets for your virtual machines, then Azure uses Zone-redundant Storage (ZRS).
- ! If you select Premium performance only LRS replication will be available.

Billing considerations

- Storage costs (per-gigabyte), data access costs, and transaction costs increases as the tier gets cooler.
- In GRS and RA-GRS, geo-replication data transfer incurs a per-gigabyte charge.
- Outbound data transfer costs (outside of Azure region) incur billing for bandwidth usage on a per-gigabyte basis.
- Changing tier from cool to hot costs as reading cool data, from hot to cool costs as writing data to cool tier.

Storing & accessing data

AzCopy

- Command-line utility designed for copying data to/from Microsoft Azure Blob, File, and Table storage

Azure Import and Export Service

- **Import:** Transfer to Azure Blob storage (**block** and **page** blobs) and Azure Files by shipping disk drives to an Azure data center.
 - **Flow**
 - a. The customer prepares the hard drives using Import/Export Client Tool (WAImportExport Tool) that facilitates the process & encrypts the drive with BitLocker.
 - b. The customer creates an import job using the Azure Portal / REST API.
 - c. The customer ships the hard drives to the data center and enters tracking number in Azure.
 - d. Hard drives are shipped back to the customer after the data is imported.
- **Export** Transfer from Azure storage to hard disk & ship to on-promise sites. You can export **Block**, **Page** and **Append** blobs from Azure.
 - Requires you to ship empty disks.
 - Costs extra egress charges when data is exported.
 - **Flow**
 - a. Customer creates export job with address & blob container path(s).
 - b. The drives are encrypted with BitLocker; the keys are available via the Azure portal.
- When?
 - Uploading/downloading is slow or costs much

- E.g. migrating data to the cloud, content distribution to customers, backup, data recovery
- ! Only specific types of disks are supported.
- ! Many regions are not available (US only now).

Security

Network configuration

- As default it accepts requests from any client, in Settings->Firewalls and virtual networks you can limit networks by choosing **Selected Networks**.
- You can then add network rule in VNet to allow access to service endpoint of the Storage.
 - When planning for disaster recovery during a regional outage, you should create the VNets in the paired region in advance.
- You can also set-up IP rules in Storage Account -> Firewalls and Virtual Networks -> Firewall -> Address range for cloud and hybrid scenarios.
 - ! Only allowed for public ip addresses

Shared access signature

- Shared access policy includes a key (signature) to be used in query string to requests to the storage.
- The policy can grant access to everything or can limit the access to specific operations (e.g. only read or writes).

Monitoring storage

- Capacity metrics
 - Sub-service specific
 - E.g. for blob: *Blob Capacity*, *Blob Container Count*, and *Blob Count*
 - Sent every hour, refreshed daily.
- Transaction metrics
 - Sent every minute from Azure Storage to Azure Monitor.
 - Available at both account and service level.
 - E.g. *transactions*, *egress*, *ingress*, *availability*
- Metrics and logs are stored in the same storage account
- Requires you to enable logging.

- ! Not supported for premium storage accounts as they work differently and for Azure virtual machines

2.3. Azure Content Delivery Network (CDN)

Azure Content Delivery Network (CDN)

- Increases speed and availability
- Caches content to the user by using servers that are closest to the users.
- Can compress (can be enabled in Azure portal).
 - Also modify the MIME types list to tune which content formats to compress.

Flow

1. User sends request to an *Edge Server*
2. DNS routes the request to the best performing Point-of-Presence (POP) location (probably geographically closer)
3. If edge does not have the content, it sends request to origin.
 - Origin can add HTTP headers describing the file's Time-to-Live (TTL).

CDN Profiles

- You can choose between Microsoft, Akamai or Verizon as implementation.
- Four choices for **Origin type**: Storage, Cloud Service, Web App, and Custom origin.
- Supports compression, query string, and geo filtering.
- Endpoint isn't immediately available for use.
 - Microsoft standard = 10 min, Akamai = 1 min, Verizon = 90 min
- Rules
 - You can set global caching rules
 - e.g. TTL Cache Expiration Duration
 - Or custom caching rules
 - If a pattern match in path & file extensions you can override global caching rules.

Optimization options

- Vendor specific settings

- **General web delivery:** Web content, website, applications, small images
- **General media streaming:** Live & video (so you don't need to change between live and VOD).
- **Video on demand media streaming:** Only for video streaming.
- **Large file download:** Large file download: > 10 MB.
- Uses chunking
 - To disallow failing of whole request.
 - Uses pre-fetch: starts fetching next chunk if the download is on the chunk before.

2.4. Virtual Machines

Virtual Machines

- IaaS (Infrastructure as a Service)

Good commands when practicing

- 🐞 Allow ping: `New-NetFirewallRule -DisplayName "Allow ICMPv4-In" -Protocol ICMPv4`
- Install IIS (Webserver): `Install-WindowsFeature -name Web-Server -IncludeManagementTools`

Grouping of VMs

- ! Defined during creation of VMs, cannot be applied to existing VMs.

Availability set

- IaaS (Infrastructure as a Service)
- Holds different tiers (instances) so Azure split them onto different places on hardware.
- Pay only for VMs
- Two logical grouping:
 - **Fault domains:** Grouping of domains with common power source + network switch. Separation protects you from hardware/network/power failures.
 - ! Max 3
 - **Update domains:** Grouping when maintaining (reboot etc) at a time.
 - ! Default = 5, Max: 20


Scale set

- IaaS (Infrastructure as a Service)
- A scale set is an implicit availability set with five fault domains and five update domains.
- Holds identical instances of images in the scale set so they can be scaled
- You deploy it via ARM template

VM Sizes






- The number of network interfaces they can support is different
- **!** Resizing VM causes downtime as the state needs to be stopped.
- Types:
 - **General Purpose:** Dsv3, Dv3, DSv2, Dv2, DS, D, Av2, A0-7
 - Balanced CPU-to-memory.
 - Ideal for dev / test and small to medium applications and data solutions.
 - **Compute optimized:** Fs, F
 - High CPU-to-memory.
 - Good for medium traffic applications, network appliances, and batch processes.
 - **Memory optimized:** Esv3, Ev3, M, GS, G, DSv2, DS, Dv2, D
 - High memory-to-core.
 - Great for relational databases, medium to large caches, and in-memory analytics.
 - **Storage optimized:** Ls
 - High disk throughput and IO.
 - Ideal for Big Data, SQL, and NoSQL databases.
 - **GPU optimized:** NV, NC
 - Specialized VMs targeted for heavy graphic rendering and video editing.
 - **High performance:** H, A8-11
 - Most powerful CPU

Connect to Virtual Machines

- Windows
 - In portal, click connect, download RDP file. (enabled by default)
 - SSH in with more configurations
-  Linux
 - Password
 - SSH in (*enabled by default*)

- Optionally you can set-up desktop environment, download and configure (*open ports in NSG and VM firewall*) RDP client and start it for RDP connections.
- SSH public key
 - Use PuTTY:
 - Go to session that you saved and type IP address there, e.g. linuxuser@12.84.169.43.
 - In connection SSH=>Auth=>Browse, select the private key.
 - Click "open"

Common management tasks

-  Moving virtual machines between resource groups or subscriptions
 - Can do using Powershell or Azure Portal
 - It'll prompt to you to move all other dependencies as well.
 - **!** You cannot move Azure Managed Disks at this time.
 - New resource IDs are created as part of the move. After the VM has been moved, you will need to update your tools and scripts to use the new resource IDs.
-  Change VM size (scale up & down)
 - **!** If size is not supported on the physical cluster it's hosted, it'll require deallocation of the VM
 - **!** If the new size for a VM in an availability set is not available on the hardware cluster currently hosting the VM, then all VMs in the availability set will need to be deallocated to resize the VM.
 - **!** Deallocation may result in Dynamic IP change.
-  Download a VM ARM template
 - Includes all of the dependencies.
 -  Storage account name must be unique so you can use something like: [uniqueString(subscription().subscriptionId)]
-  Swap OS Disk
 - You don't have to delete/recreate/de-allocate the VM.
 - If it's managed disk it must not be in use.

Monitoring

- Available both for Windows & Linux
 - Diagnostic settings (collected information types) are different on windows & linux.
- Implemented as Agent VM extension.
- Configure after/during deployment:



- **Boot diagnostics**
 - To see why VM gets into a non-bootable state
- **Guest OS diagnostics**
 - Collect additional disk, CPU, and memory data.
- Gathers metrics, diagnostics, and log data
 - Displayed in Azure portal (metrics and boot diagnostics)
 - Stored in Azure storage of a dedicated account (tables and blobs)
- **Azure Network Watcher** is a blade in Azure that provides tools to monitor, diagnose, view metrics, and enable or disable logs for resources in an Azure virtual network.
 - You can add ***Connection monitors** with source and destination VM to track connection between them.
- **💡** If you have many different VM machines (e.g. on different tiers), consider larger scale distributed application monitoring systems such as Microsoft EMS.
- **Network Watcher** enables you to monitor your VM and its associated resources as they relate to the network that they are in.
 - You can install the Network Watcher Agent extension on a Linux VM or a Windows VM.

Azure Disks vs Azure Files vs Azure Blobs

- Always hosted in a Storage account.
- **Azure Files**
 - Good for "lift and shift" operations.
 - Good if data will be shared by multiple applications.
- **Azure Blobs**
 - Support for streaming & random access scenarios.
 - Access application data from anywhere.
 - **!** Capacity: Max 2 PiB Account Limit
 - E.g. enterprise data like to perform big data analytics.
- **Azure Disks**
 - Handles storage account creation for you.
 - Good for lift & shift scenarios for applications that use native file system APIs.
 - **!** Capacity: Max 5 TiB file shares.
 - You want to store data that's not required to be access from outside the VM to which the disk is attached.

2.4.1. Virtual Machines - Deploy



Deploy Virtual Machines

-  Redeploying (button in VM) a VM changes its host.
 - Causes downtime
- Methods: Azure Portal, ARM templates, Azure Powershell, Client SDKs, Rest APIs, Azure CLI, Visual studio
-  In order to qualify for 99.95% SLA, you need to deploy two or more VMs running your workload inside an availability set.
- Choosing location of VMs
 - Region: Regional pairs allows replication resources, e.g. VM storage
 - Feature availability might change per region



Steps using portal

1. Select image or disk.
 - **Image**
 - Sources
 - **Azure Marketplace**
 - Recent versions of Windows Server & Linux distributions.
 - Some images contain applications, e.g. SQL Server.
 - Linux images are created with Bitnami and certified for Azure
 - **VM Depot**
 - Community managed repository of Linux and FreeBSD virtual machine images.
 - **Custom Images**
 - Create and upload for use in Azure
 - Types
 - **VM Image**
 - Includes an operating system and all disks attached.
 - **OS Image**
 - .vhd file with generalized (via sysprep tool) version of OS.
 - No additional disks are attached.
 - **Disk**
 - VHD => Virtual Hard Disk
 - You create your own, or use disks from back-ups.
 - Disks used by VMs
 - **OS disk**
 - Has OS e.g. C: drive

- **! Max 2 TB**
 - **Temporary disk**
 - E.g. D: drive
 - Short term storage
 - Not consistent.
 - **Data disk**
 - Stores application data.
 - **! Max 4 GB, if it's managed 32 GB.**
 - Disk choices
 - **Managed**
 - Storage is hidden from you.
 - Microsoft recommended
 - Granular access control
 - Azure Backup service support
 - Better reliability for availability sets
 - Ensures in a set they're isolated from each other.
 - No single point of failure.
 - **! Up to 10.000 VM disks in a subscription.**
 - **Unmanaged**
 - You connect a storage account to connect a page blob storage
 - **💡 Careful by not having many VMs in same storage account. You can go over the limits!**
2. Provide required information
- Such as host name, user name, password for the virtual machine.
 - Authentication types in Linux:
 - **Password**
 - Using passwords with SSH connections still leaves the VM vulnerable to brute-force attacks or guessing of passwords.
 - **SSH public key**
 - More secure & preferred method
 - SSH is an encrypted connection protocol that allows secure logins over unsecured connections.
 - **Public key:** The public key is placed on your Linux VM, or any other service that you wish to use with public-key cryptography.
 - You can change your public key later on.
 - You need RSA public key, it can be generated using ssh-keygen on Linux and OS X or PuTTYGen on Windows.
 - **Private key:** The private key is only for you.

-  Azure currently requires at least a 2048-bit key length and the SSH-RSA format for public and private keys.
- 3. Provide optional information
 - E.g. domain membership, virtual networks, storage account and availability set.
 - Availability Sets
 - VMs in same set placed at different physical hardware to help prevent downtime by Azure maintenance or regular cycles or failure.
 - if there are multiple VMs in same set they don't go down at the same time.
 - Auto-shutdown
 - When the VM is shut down automatically.
 - With "Azure Automation" and have it boot & shutdown on regular basis.
 - Network security group
 - Firewall where inbound/outbound rules lies.
 - During creation you can set:
 - E.g. HTTP/RDP in a checklist.
 - E.g. SSH for linux
 -  Default is all outbound traffic is allowed.
 - RDP for windows is allowed per default for outbound ports.
 - Extensions: E.g. Microsoft Antimalware
 - Monitoring:
 - **Boot:** Records boot sequences, screenshot, error logs etc.
 - **Guest OS diagnostics:** about the OS where the application is running

Deploy custom images

-  Several options
 - Use an existing managed disk.
 - Upload a VHD.
 - Copy an existing Azure VM by using snapshots
-  Deploy through uploading VHD
 - i. Prepare VHD on-premises
 - Make sure the virtual machine has all the roles and features installed that you need.
 - Generalize
 - Be sure to remove any guest virtualization tools and agents.

- Ensure the VM is configured to pull its IP address and DNS settings from DHCP. This ensures that the server obtains an IP address within the virtual network when it starts up.
 - **sysprep**
 - Generalization tool for e.g. computer name, security identifier (ID) (*critical in AD domains as logins and SIDs are tied together*), driver cache, other unique ID's.
 - It generates a VHD.
 - Prepare VM VHD
 - Azure accepts only VHD, use utility tools to convert VHDX, VMDK to VHD.
- ii. Create the Storage Container and upload the VHD

2.4.2. Virtual Machines - Azure Backup & Azure Site Recovery & Snapshots

Backups

Azure Backup

- Managed service for back-ups.
- Use for production workloads.
- Creates recovery points that are stored in geo-redundant recovery vaults.
 - Allows you to restore whole VM or specific files later on.

Setting up

1. Create a recovery services vault
 - It stores your back-ups and recovery points.
 - Choose between geo-redundant (default) or locally redundant.
2. Define the backup in vault.
 - Snapshots (recovery points) are stored in the vault.
 - You can restore VM from any recovery point.
 - Create new back-up
 - Set backup goal with
 - Source: From where (Azure/on-premises/Azure Stack)
 - Object: What e.g. VM, SQL (depended on source)
 - 💡 In Site recovery the action is called **protection goal**

- Configure back-up policy:
 - Frequency (retention range): How often? Ex. once a day, weekly, yearly etc.
 - When exactly? (ex. 11:00 AM Sunday)
 - You can back-up directly with "Back-up Now"
- 3. Back-up virtual machine
 - Azure VM Agent must be installed on VM, they're already installed for VMs from gallery.
 - Backing up VMs is within region.
 - **!** You cannot back up VMs from one region to a Recovery Services vault in another region.
 - **!** For every Azure region that has VMs to be backed up, at least one Recovery Services vault must exist in that region.
 - For on-premises, you need to install VM agent

Restoring

1. You can choose to restore only files.
 - Restore types: [Three Ways to Restore Azure IaaS VMs](#):
 - a. **Create a virtual machine**
 - Basic settings such as VNet, subnet and storage account.
 - **!** Does not support advanced settings such as e.g. VMs under load balancer, multiple reserved IPs or multiple NICs.
 - b. **Restore disks**
 - Copies VHDs into a storage account you specify.
 - You can then create a new VM using these disks or attach existing VM to the disk.
 - c. **Direct File Recovery**
 - Select recovery point -> Download script to mount VM disks so you can browse them -> Once you restore files, unmount the drives.
2. Create a new VM
3. Create a new storage disk (copies VHDs).
4. Replace existing VM
 - **!** Only supported for unencrypted managed VMs from marketplace).

Azure Site Recovery

- Protects from a major disaster scenario when a whole region experiences an outage.
 - E.g. due to major natural disaster or widespread service interruption.

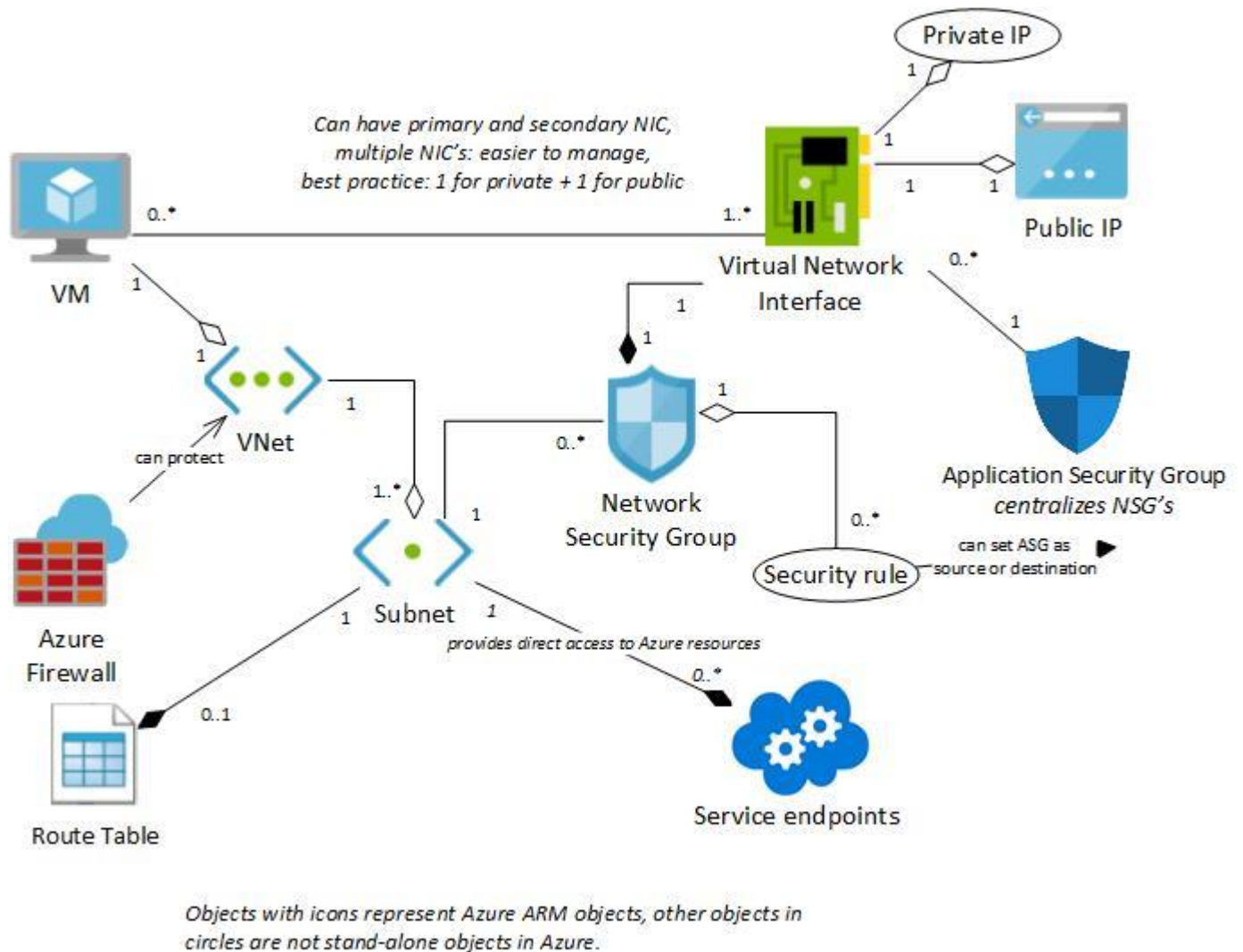
- You can replicate to an Azure region of your choice.
- You can set up easily on Portal => VM => Disaster recovery with target region.
- Explained more [here](#).

Snapshots

- Read only-copies of managed disks.
- Provide a quick and simple option for backing up VMs.
- Snapshot for consistency requires you to stop VM, Azure backup handles it via an extension
- Can be used to rebuild VM / create new managed disks as they exist independently.
- Billed based on the used portion of the disk (not whole disk capacity).

2.5. Virtual Networks

Virtual Networks



- IaaS (Infrastructure as a Service)
- Virtual networks are software defined logical isolation that mimics the hardware.
- Virtual networks are defined with address space and at least 1 subnet.
 - ! Use an address space that is not already in use either on-premises or in other VNets.
 - If it overlaps => you will have to reconfigure or recreate the VNet.
 - How many to choose? Be careful!
 - ! A best practice: "one octet further in"
 - E.g.: VNet (11.0.0.0/ 8) => Subnet (11.1.0.0.0/ 16)
- ! Limitations:
 - By default, you can create up to 50 virtual networks per subscription per region, you can contact support to increase it to 500.

- A virtual network and its resources are scoped to a single region and single subscription.
 - However, different VNets from different regions can communicate through **global peering**.
- When you move VNet to another resource group, you must move all of its dependencies.
- Name resolution in a VNet:
 - i. Azure-provided (*default*)
 - ii. Custom DNS server
 - E.g.: Deploy a DNS server and type its address.

Filter network traffic

Security groups

- Control inbound / outbound rules
- Network Security Groups
- Application security groups (on application level)

Network virtual appliance (NVA)

- A network virtual appliance is a VM that performs a network function, such as a firewall, WAN optimization, or other network function.
- They're often used in DMZ (perimeter network)
 - **DMZ**: Security zone protects internal network from an untrusted network.

Routing

Route tables

- Associated to one or more subnets.
- Defines routes = where next hop will go
 - If matching route can't be found, the packet is dropped.

System routes

- Azure manages following situations with **system routes**:
 - VM <=> VM in same subnet

- VM <=> VM in different subnet in same VNet
- Internet <=> VM
- VNet-to-VNet VPN communication between VMs
- Site-to-Site and ExpressRoute communication through the VPN gateway.

Custom routes

User-defined routes (UDRs)

- To a virtual appliance (router, firewall, WAN optimization)
- Appliance between subnets tier (front-end back-end) or (subnet internet)
- 💡 Deploy virtual appliance to different subnet. If it's in same subnet where it's routed it can result in routing loops where traffic never leaves the subnet.

Border gateway protocol (BGP)

- Standard protocol commonly used.
- Routes are automatically added to route table.
- Obligatory with Express Route, optional with VPN Gateway
- Support multiple tunnels.
- Provides transit routing between on-prem <=> Azure VNet via VPN gateway.
 - Transit route: A->B & B->C => A->C
- You can propagate your on-premises BGP routes to your virtual networks through Azure VPN Gateway or ExpressRoute connection.

How Azure selects a route

- Routing algorithms:
 - If address matches multiple routes then Azure makes decision based on:
 - a. **Prefix match algorithm**: Less range is selected.
 - b. **Route priorities** (if prefixes are the same)
 - User-defined route > BGP route > System route
- 0.0.0/0 address prefix => Routed to internet by default (can be overridden).

Azure Gateways

- They're used to exit & come back in to private network.
- Requires gateway subnet to exist to the VNet

- ! Do not associate NSG on gateway subnet.

Azure Application Gateway

- Layer 7 load balancers / reverse proxies
- ! Works with different availability zones but only across same region.
- Supports:
 - **SSL termination**: Traffic flows unencrypted to the back-end servers.
 - **Connection draining**: Remove back-ends using REST or they're removed automatically if their health status is not ok.
 - **Custom error pages, Web application firewall, URL-based routing, multiple-site hosting, redirection, session affinity, websocket, rewrite HTTP headers**

Azure Virtual Network Gateway

- It's assigned to a VNet in a gateway subnet.
 - A VNet can have 1 gateway subnet and only 1 gateway in it.
- Used for Azure <=> Azure or Azure <=> On-prem
- Traffic goes through public internet
 - Requires public IP.
- Two types: VPN, ExpressRoute
 - **Azure VPN Gateway**
 - Encrypts & routes the traffic via IPSec/IKE (*better than SSL*)
 - Between two or more virtual machines in **gateway subnets**.
- 🗑️ Types:
 - **PolicyBased**
 - Also called **static routing**
 - IPSec policies decides where/whether to send.
 - **Route based**
 - Route table decides where/whether to send.
 - Also called **dynamic routing**
 - Required for **Multi-Site VPN, VNet to VNet**, and **Point-to-Site**.
- **SKU**
 - Limits the number of tunnels you can have and the aggregate throughput benchmark.
 - ! Basic has no public IP.

2.5.1. Virtual Networks - Virtual Network Connectivity

Virtual Network Connectivity

Communicate between Azure resources

- **Through virtual network service endpoints**
 - Endpoints allow you to secure your critical Azure service resources to only your virtual networks.
 - Available for: • Azure Storage • Azure SQL • PostgreSQL • MySQL • Cosmos DB • Key Vault • Service Bus • Event Hubs
- **Through a virtual network**
 - Some resources can be deployed directly to a virtual network.
 - E.g. Redis, Azure Kubernetes Service, App Service Environment...

Communicate with on-premises resources or intersite connectivity

VPN Gateway Connections

- Gateway to gateway connections.
- Requires shared key that both parts know.
- Azure VPN gateways provide secure tunnel using IPSec/IKE.
- You can see & verify established connections in VNet → Gateway → Connections blade as "Status: Connected"

VNet <=> VNet

- Works across regions, subscriptions, deployment models, cloud/on-prem.
- 💡 Use one VNet as gateway and peer other ones on Azure. Gateways on every VNet is costly & slow!
- On Azure the connection does not go over the internet.
- Deploy Gateway on each VNet.
 - Set-up connection in VNet1 gateway to VNet2
 - Set-up connection in VNet2 gateway to VNet1

Site-to-site (S2S)

- If one VNet is on-prem it's called **Site-to-site (S2S)**
- On-premises VPN device <=> Azure VPN Gateway
- The local gateway is configured manually
- Problem: All users download VPN client to connect Azure.
 - Easier: Have a hardware device as jumpbox and use it as gateway through secure ip tunneling.
 - Even easier: ExpressRoute.

VNet <=> Device (with VPN Gateway)

- Or **Point-to-site virtual private network (VPN)**
- Configure =>
 - IP address space for clients.
 - Configure virtual gateway
 - Create root and client certificates & upload to azure
 - Install VPN client configuration created by Azure
 - Connect to VPN

Azure ExpressRoute

- Private connection that does not go over the internet.
- BDP is the only routing way.
- Reliable => Circuits consist of two connection to Microsoft Enterprise Edge.
- Facilitated by a connectivity provider (e.g. Telia, Tele2)

Virtual Network peering

- The virtual networks you connect are across subscriptions and regions.
- All traffic is routed over Azure internal networks, handled by Azure infrastructure.
- Faster & easier to setup than VPN
- No public IP required
- No downtime when creating/configuring peering.
- Regional network peering => In same VNet
- Global network peering => Cross region VNets (in preview)
- Requirements:
 - Public clouds (not Azure national clouds)

- Resources in one VNet cannot communicate with load balancer in the peered VNet. Load balancer and resources that communicate with it must be in same VNet.
- **! Limitations:**
 - Not transitive.
 - E.g.: VNet1 <=> VNet2 and VNet2 <=> VNet3 does not mean VNet1 <=> VNet3
 - No overlapping address spaces
 - Any address space changing => Destroy peering re-deploy

Virtual Network peering settings

- **Allow forwarded traffic**
 - Allows traffic not originated from within peer VNet to VNet.
- **Allow gateway transit**
 - Peer VNet uses your network gateway.
 - Allows you to have single gateway, instead of gateway per VNet (called hub-spoke topology)
 - Examples:
 - Site-to-site VPN <=> On-prem network
 - VNet-to-VNet <=> Another VNet
 - Point-to-site VPN <=> Connect client
- **Use remote gateways**
 - Uses peer's virtual gateway.

2.5.2. Virtual Networks - Virtual Network Interface

Virtual Network Interface

- Groups & manages public + private IP.
 - The address for each NIC are within the same subnet.
- Associations:
 - A VM must have at least one network interface attached to it.
 - It can have **network security group** associated with it.
 - It has a **VNet** and **subnet** associated with it.
- **💡** Adding an virtual network interface does not cause any downtime.

Multiple NICs


- Best practice recommended by Microsoft.
- Needed for many network virtual appliances.
- ⚠ Having different front-end and back-end NIC(s) makes administration/management easier
- **Primary NIC** is first NIC attached, **secondary NIC(s)** are the others.
- By default outbound traffic is sent by IP from primary NIC and Load Balancer pool uses primary NIC IP.
- ! Limitations
 - VM size limits how many NICs can be attached.
 - Only one NIC can have public IP.
 - The order (*names*) of the NICs inside the VM will be random or can be changed after Azure updates etc, but IP and MAC addresses stay the same.
 - In an availability set for each VM must use either multiple NICs or single.
 - You cannot mix.
 - Single NIC <=> Multiple NIC(s) configuration requires VM to be re-deployed.

IP addressing

- **Prefix:** e.g. 10.1.0.0/24
- ! Dynamic <=> static switch requires NIC to be restarted.
- Effects subnet configuration.
 - Azure best practice is to manage subnets separately.
 - Subnet for static IPs and subnet for dynamic IPs.
 - E.g. in multi-tier application web servers and load balancers will have public IPs but internal web application data layers won't have public IPs.
 - E.g. in big infrastructure where they have one or two jump boxes that have public IPs for the purpose of doing administration.
 - A box you can RDP to and then from there access other systems inside the implementation versus giving everything a public IP.
- Default gateway is completely managed by Azure. You cannot modify.
- You can set custom DNS server.
 - DNS server of VM is inherited from VNet, not IP address.

Public IP

- Used for external internet communication.
- Azure ARM object with a globally unique name.
- Used in: • VMs • load balancer • VPN gateway • Application Gateway.

- Can be static or dynamic.
 - Static IP do not change and is good for:
 - DNS name resolution.
 - IP address-based security
 - SSL certificates linked to an IP
 - Firewall rules
 - Role-based VMs such as domain controllers and DNS servers
-  SKUs
 - Basic
 - Can be assigned to any Azure resource
 - Assigned to a zone and not zone redundant.
 - Standard
 - Always static.
 - **!** Can only be assigned to: NICs, public standard load balancers
 - Zone redundant by default.

Private IP

- Used within VNet and subnets.
- Can be used on-premises with VPN gateway or ExpressRoute.
- Can be static or dynamic
- Resources: in VMs, Load balancers and Application gateway.

IP Forwarding

- Modifies IP address to reach right target.
- Allows transient flows. E.g. NIC3 lets a NIC1 trying to communicate with NIC2 that it has no route to but only to NIC3 by IP forwarding during routing.

2.6. Azure Migration

Azure Migration

Migration Phases

Discover

- Inventory of on-prem resources to plan where the migration should start
- Tools: **Azure Migrate Service, Database Migration Assistant**
- Answer to questions such as _What are my applications? How are they made up JAVA or .NET? Data structure, SQL VM or SQL? How will they look like in Azure?
- You can use Azure partner discovery services such as Cloudamize, CloudSpace...

Migrate

- Tools: **Azure Site Discovery, Azure Database Migration Service, Azure Data Box**
- Deploy identity, network, storage, and compute infrastructure
 - You move selected workloads to Azure.

Optimize

- Fine tune your Azure-based workloads and maximize your ROI (Return on Investment).
- Tools
 - **Azure management and security**
 - E.g. • backup • monitoring • security • assessment
 - **Azure Cost Management**
 - Create budgets and alerts with spending thresholds
 - Cost tracking, analysis
- Security + performance improvements
- 3rd parties help with with backup, monitoring, security assessments, and cost management.

Arguments for migrating

- No hardware obsolesce cycle: No need to sell hardware after a while
- No pre-purchase capacity model, but pay for what you use.
- Lack of IT agility
- Desire to focus on core competencies
- Expense of maintaining a global presence
- Enable disaster-recovery scenarios: Geographically dispersed locations.

2.6.1. Azure Migrate Service

Azure Migrate Service

- Free tool for primarily IaaS-based assessments.
- Good for lift-and-shift migrations.
- Supports VMware-virtualized Windows and Linux VMs.
- Non-intrusive discovery of on-premises VMs & workloads
- Examines & assets:
 - **Azure readiness**
 - Suitability of on-premises machines
 - Asserts • ready for Azure, • ready with conditions, • not ready for Azure, • Azure readiness unknown (when readiness cannot be identified due to data unavailability)
 - **Sizing suggestions**
 - For VMs & disks based on history
 - Two settings:
 - As on-premises
 - Performance based
 - Based on utilization history
 - *Storage*: default is Premium disks
 - *Network*: performance required by network adapters
 - *Compute*: CPU & memory requirements
 - **Cost estimation**
 - The estimated cost for running the machines & storages in Azure
 - **High confidence migration**
 - Migration risks and recommended tools: recommends e.g. **Azure Site Recovery**
 - Visualize dependencies of on-premises machines through **dependency maps**
 - Create groups that you will assess and migrate together
- Assessment content
 - Target location, Storage type, Reserved Instances, Sizing criterion, Performance history, Comfort factor, VM series, Currency, Discount (%), VM uptime, Azure offer, Azure Hybrid Benefit
 - **Comfort factor**: Buffer that's applied on top of machine utilization data for VMs.
- **!** Assesses only VMware (>5.5) environments, for Hyper-V machines use *Azure Site Recovery Deployment Planner*.

Flow

1. Create migration project
 - In Azure, create an Azure Migrate project.
2. Install **Collector**
 - You download .OVA & import in VMware vCenter as VM
 - Read-only VM to log
3. Configure **Collector**
 - You connect to console of VM or web to initiate the discovery
 - Copy & paste your project id and key from Azure.
 - It reads: config data, virtual processors, memory size, disk, network configuration, performance history (CPU utilization, memory, disk IOPS & throughput, network output to choose right size for VMs)
4. Select VMs or groups (can customize groups) & create assessment.
 - Customize machines in report to recalculate costs.
5. You can optionally install **Dependency Agent** to see dependency maps

2.6.2. Azure Site Recovery Service

Azure Site Recovery Service




- DRaaS: Disaster Recovery as a Service
- Tool for VM replication and migration from anywhere to Azure
- Containerization of existing applications and infrastructure.
- Modernization options for apps and databases.
- Can automatically replicate to Azure or from Azure
 - Supports Hyper-V, VMWare, Windows, Linux VMs
- Pre-, post scripting with **Azure Automation**
 - Safeguard complex workloads against outage with
- Use-cases
 - Use Azure as a secondary site for conducting business during outages.
 - Continuous health monitoring remotely from Azure.
 - Lifting and shifting of servers, apps, databases, and data.
 - **Lift and shift:** A strategy for moving an application or operation from one environment to another without redesigning the application.
- Pricing
 - First 31 days free then to Azure (25\$), to on-prem(15\$) per month for single instance

- Storage always costs
- Azure Compute Charges for recovered VMs (if fail-over / migration is done)
- Outbound data transfer (egress charges, when failing back)
- ***Azure Backup vs Azure Site Recovery**
 - You need to have both to have a full business continuity plan.
 - **Azure Backup:** Copy of data to restore business back to a specific period of time

Azure Site Recovery Deployment Planner

- A command line tool that gives an excel file
- Includes on-premises summary, recommendations, VM storage placement, compatible + incompatible VMs, on-premises storage requirement, initial replication batching, cost estimation

Setting up site Recovery

- Prerequisites:
 -  Minimum permissions: Virtual Machine Contributor, Site Recovery Contributor, write to selected storage account.
 - Create storage account to store replicated VMs
 - Create VNet
 - Plan with [Azure Site Recovery Deployment Planner](#)
-  Steps:
 - i. Set up **Recovery Services vault**
 - Create a resource -> Management Tools -> Backup and Site Recovery
 - It stores your back-ups and recovery points.
 - Backups are protected with Azure Backup (prevention, alerting)
 -  Create or manage a Recovery Services vault **in the context of the target service.**
 - Central monitoring: Azure VMs + on prem, RBAC
 - **Azure Recovery Services vs Back-up Vault**
 - You can no longer create Backup vaults, and all existing Backup vaults have been upgraded to Recovery Services vaults.
 - ii. Set up target environment in Azure
 - Select storage, recovery vault and network
 - LRS or GRS storage is recommended, so the data is resilient if a regional outage occurs, or if the primary region cannot be recovered.
 - iii. Select a replication goal (**protection goal**)

- Recovery Services vaults > vault > Recovery > Prepare Infrastructure > Protection goal
- Select what you want to migrate e.g.
 - **VMware**: Select To Azure > Yes, with VMWare vSphere Hypervisor.
 - **Physical machine**: Select To Azure > Not virtualized/Other.
 - **Hyper-V**: Select To Azure > Yes, with Hyper-V. If Hyper-V VMs are managed by VMM, select Yes.
- iv. Set up the source environment
 - Download and import ASR configuration server and process server into vCenter Server
- v. Replicate the VMs that you want to migrate to Azure
 - You can create **replication policy**
 - You set how often recovery points will be created.
 - Recovery Services vault -> Site Recovery infrastructure > Replication Policies > +Replication Policy.
 - Enable replication directly
 - In the vault, click +Replicate.
 - Select VMs
 - Everything is copied in storage.
 - **! Paging files** have a lot of churn and will generate a lot of replication events.
 - You can optimize via:
 - Split the single virtual disk into two virtual disks. One virtual disk has the operating system, and the other has the paging file.
 - Exclude the paging file disk from replication.
 - **Pagingfile**: a pagefile is a reserved portion of a hard disk that is used as an extension of random access memory (RAM) for data in RAM that hasn't been used recently.
- vi. Run **test failover**: it's *no impact migration testing*.
- vii. Switch production environment to Azure
 - Create a **recovery plan**
 - Recovery plan describes the way migration will work in the event of a disaster.
 - In recovery plan, you can:
 - Change order of VM starts.
 - Customize fail-over
 - Run scripts via **Azure Automation**
 - **Azure Automation**
 - Runs runbooks (piece of script of NodeJs, PowerShell, Python)

- Has support for modules.
- Select order
 - Create groups
 - VMs in same group are shut down and booted in the same time
 - Across groups they're booted & shut down sequentially, without groups at the same time
 - Choose VM size and customize other settings by clicking on "*replicate items*" in *recovery plan*.
- Set the target IP address.
 - 💡 If you don't provide an address, the failed-over machine uses DHCP.
- You can use Hybrid Use Benefit (up to 40% cut for existing Windows Server licenses)
- **Failover** to azure on recovery plan
 - Settings > Replicated items -> Click the machine > Failover.
 - **Failover & Failback**
 - The **failover** operation is the process of switching production to a backup facility (normally your recovery site)
 - Failover isn't automatic but a manual process.
 - **Unplanned failover**
 - E.g. natural or IT disaster.
 - Done from latest point with minimal data loss
 - **Planned failover**
 - You choose the point, zero data loss
 - A **failback** operation is the process of returning production to its original location after a disaster or a scheduled maintenance period.
 - Failback is a planned failover from Azure to on-premises
 - **Flow:** Go to vault -> Click on "planned failover" -> choose data synchronization -> Choose between a full download (quicker) or synchronization of delta changes (lesser downtime)
 - Two ways to synchronize data:
 - **Full download:** A full download is faster but requires the VM to be shutdown which causes more downtime.

- **Minimize downtime:** More time to synchronize the changes, but the VM is not shut down so less downtime.
- After failover, ***Commit migration for production***
 - Choose VMs for fully migration, click on "Complete migration" on recovery plan.

From on-premises to Azure

1. Set up an Azure storage account
2. Create a vault
3. Select a protection goal: To Azure > Not virtualized/Other
 - ! In Azure Backup it's called Backup Goal
4. Set up the source environment
 - Recovery > Prepare Infrastructure > Source > +Configuration server > Add Server
 - Download the vault registration key
 - Download & install the **Site Recovery Unified Setup** installation file.
5. Set up the target environment: Prepare infrastructure > Target
6. Create a replication policy to choose when (how often) to replicate
 - Site Recovery infrastructure > Replication Policies > +Replication Policy.
7. Enable replication
 - ! Limitations:
 - 64 bit only.
 - Max disk size: 4TB
 - Max OS size: 2TB but for generation 2 hyper-v it's 200GB

3.1.1. Identities - Azure Active Directory

Azure Active Directory

- Different than Active Directory, it's flat: no domains, objects, forests etc.
- ***Identity management solution***: multi-factor authentication, device registration, self-service password management, self-service group management, privileged account management, RBAC, application usage monitoring, auditing & security monitoring & alerting.
- Tenant in Azure = AAD instance

- Each subscription trusts an AAD tenant, multiple subscriptions can trust a single tenant.
 - Add custom domain
 - All AD accounts come with domainname.onmicrosoft.com
 - Go to AD -> Custom Domain Names -> Add domain -> Verify domain name with a TXT record
- Used by (can be shared with) Office 365, Microsoft Dynamics CRM and Microsoft Intune.
- List of users + groups
- Each Azure tenant has a dedicated and trusted Azure AD directory. The Azure AD directory includes the tenant's users, groups, and apps
- HTTP/HTTPS based:
 - Queried via REST using Microsoft Graph API
 - Uses protocols like SAML, WS, OpenID and OAuth 2 for authorization
- Other benefits
 - Single sign-on to any cloud (Office365, salesforce, dropbox, custom ..) or on-premises web app.
 - **Self-service tools** (means tasks not requires an admin):
 - **Self-service password management:** Users can reset their passwords
 - Authentication methods:
 - Email, mobile phone, office phone, security questions.
 - At least one must be selected, better with multiple
 - **Self-service group management:** Users can create groups
 - Enabled in Portal -> AD -> Self-Service Password Reset
 - Can be integrated with Windows Server Active Directory
 - Used in Office 365, Azure and Dynamics CRM online.
 - **Conditional access policies**
 - When this happens, Then do this
 - (Occurrences) e.g. unknown device, e-mail user
 - (Actions) **Risk-based policies:** E.g. risk is high that user is a bad actor, enable MFA (multi factor authentication)
 - Configure through Portal: AD -> Conditional Access -> New Policy
 - **Device Management:**
 - **Register** a device (can be combined with mobile device management)
 - It's cloud only solution for personal devices
 - **Join** a device = extend registered device, add corporate network so people can log-in from there.
 - For devices that are owned by your organization, can be hybrid
 - No additional authentication prompts
 - **Enterprise State Roaming (ESR)**

- Sync Windows 10 devices with Azure AD.
- Provides users with a unified experience across their Windows devices and reduces the time needed for configuring a new device

Azure AD Identity Protection

- Can be activated it from Azure Marketplace.
- Features:
 - Get view of flagged users & risk events detected by ML algorithms.
 - Set risk-based Conditional Access policies to automatically protect your users.
 - Improve security posture by acting on vulnerabilities.

Access reviews

- It works by creating an access review with defining who'll review it, what'll be reviewed (resource roles, groups, apps, directories, user(s)), and when it'll be reviewed (frequency or deadline) then assign it to a reviewer. The reviewer will then get a deadline and notification for the review.
- Good when:
 - Too many users in privileged roles
 - When a group is used for a new purpose
 - Business critical data access
 - To maintain a policy's exception list
 - Ask group owners to confirm they still need guests in their groups

Roles

- Each role is a set of properties defined in a JSON file per API.
- Includes name, id, desc + allowable permissions (Actions), denied permissions (NotActions), and scope (read access etc.) for the role.
- Built-in roles:
 - **Owner** has full access to all resources including the right to delegate access to others.
 - **Contributor** can create and manage all types of Azure resources but can't grant access to others.
 - **Reader** can view existing Azure resources.

Azure Identity and Access Management Solutions (IAM)

- Roles tab you can see and manage what the roles are and how many users/groups are assigned to the role.
- It can be on subscription/resource or Azure level.
 - Assignments are inherited down resource hierarchy: Azure > Resource Group > App

Azure role-based access control (RBAC)

- **Role assignment:** Grant access by assigning a *Security Principal*, a *Role* at a *Scope*
 - **Security principal:** User, group or service principal (resources e.g. VM, web app).
 - Service principals are assigned to a **Managed Identity**.
 - Roles and permissions can be delegated to service principals by modifying value of their Managed Identities.
 - **Role** : Built-in or custom role
 - Roles are specific to level, app type (*VM, storage*)
 - **Scope** : Subscription, resource group or resource

SKUs

- **Basic:** cloud centric application access
 - self-service identity management, group-based access management, self-service password, basic reports, object limitations, smart lockout (not configurable risk level).
- **Premium P1**
 - Better hybrid support, seamless sign on, company branding
- **Premium P2**
 - Advanced reports, write-back
 - **Identity protection:** Uses ML in background scores the risk of authentication.
 - **Privileged identity management**
 - Allows users to elevate their privilege to a higher level to perform a particular task.
 - Users elevate their account to admin to perform a task and then de-elevate the account.
 - Require Multi-Factor Authentication
 - Access reviews
 - Schedule activations for a specific date
 - Receive notifications when users are assigned

- Configure and resolve alerts for privileged roles

3.1.1.1. Identities - Azure Active Directory - Hybrid Identities

Hybrid Identities

- **Hybrid identity:** Azure identities with on-premises AD connection

Azure AD Connect

- Installed on a domain controller, handles sync to Azure AD.
- **Configuration:** Register Azure AD account and AD DS account, and do some checkboxes => done.
 - **Device writeback** Devices must be located in the same forest as the users.
- Can set up **Seamless SSO**
 - Works with both [Password Hash Synchronization](#) and [Pass-through authentication](#)
 - Automatically sign in user with their Azure ID if they're on domain joined device.
 - Works via JS, trusts Kerberos ticket.
 - Flow
 - a. **Prerequisites:** Set up your Azure AD Connect server, Enable modern authentication
 - b. **Enable the feature:** Enable Seamless SSO through Azure AD Connect.
 - c. Roll out the feature:
 - Add <https://autologon.microsoftazuread-sso.com> by using Group Policy.
 - Because browsers will not send Kerberos tickets to a cloud endpoint, like the Azure AD URL, unless you explicitly add the URL to the browser's Intranet zone.
 - d. Test the feature
 - e. Roll over keys
 - Periodically roll over these Kerberos decryption keys - at least once every 30 days.
 - Keys can be used to create tickets and are vulnerable.

Azure AD Health

- Monitors AD FS servers, AD connect, AD domain controllers.

- Synchronization between AD DS and Azure.
- Alert can be set up.

Azure authentication methods

- !Hard to change later on
- 💡 Microsoft recommendation: [Cloud only](#) > [AD Connect cloud accounts](#) > AD Connect + [PHS \(and SSO\)](#) > AAD Connect [PTA \(+ SSO\)](#) > AD Connect + [AD FS](#)

Cloud only

- No infrastructure on premises.

AD Connect cloud accounts

- Just username is synced from on-prem AD DS.
- Requires users manage two passwords.

Password hash synchronization (PHS)

- Synchronizes hashes of passwords from an on-prem AD to a Azure AD.
- Azure-based authentication security:
 - **Smart lockout:** prevents brute force attacks, and prevents genuine users from being locked out.
 - **Leak credential report:** Microsoft scans web and dark web for leaked accounts & requires password reset if account is leaked.
- Active password write-back or password changes on Azure will not be updated on-prem

Pass-through authentication (PTA)

- Authentication is on premises.
- You deploy two or more (for availability) authentication agents on-premises.
 - They make persistent connection to Azure (ExpressRoute has circuits for this)
 - Outbounds to HTTPS 443 (no need for perimeter network)
 - Pulls authentication requests
 - Pulls rules
 - When to change password?
 - What hours I can log in?

- Azure AD encrypts with public key, places it on queue, auth agent listens & decrypts with private key, authenticates and returns back the success/fail message.
- Supports **smart lock-out** but no **leak credential report**.
- Requires installation agent on the same server as Azure AD connect.
 - Install on more servers (can be domain controllers) for high availability.

Federation (AD FS)

- Federation is a collection of domains that have established trust.
- Recommended if something Azure does not support is used, can be smart card, biometrics, third party authentication etc.
- Create Azure VM domain on-prem which is a replica of domain controller in Azure.
 - You can then join machines to this managed domain using traditional domain-join mechanisms.
 - Complex set-up
 - When?
 - Lift & shift workloads: Moving all of your domain controllers to Azure.
- 2 or more (for availability) AD FS servers to accept authentication requests.
- An AD DS instance is registered in Azure.

3.1.2. Identities - Active Directory Domain Services (ADDS)

Active Directory Domain Services (ADDS)

- Other name: Windows Server Active Directory
- On prem solution.
- Different architecture than Azure Active Directory.
- Logical divisions:
 - **Objects**: users, printers etc.
 - **Domain**
 - Groups objects
 - Each domain holds a database containing object identity information.
 - Domains are identified by their DNS name structure, the namespace.
 - **Tree**
 - A collection of one or more domains and domain trees in a contiguous namespace
 - Linked in a transitive trust hierarchy

- **Forest**
 - At top of the structure
 - A collection of trees that share a common global catalog, directory schema, logical structure, and directory configuration.
 - The forest represents the security boundary within which users, computers, groups, and other objects are accessible.
- **Domain controller (DC)** is a server computer that responds to security authentication requests (logging in, checking permissions, etc.) within a domain.
 - Multiple instances can be deployed.
- You can deploy AD DS to Azure as VM but:
 - You manage the deployment, configuration, virtual machines, patching, and other backend tasks.

Active Directory Federation Services (AD FS)

- Included by Active Directory Domain Services (ADDS)
- Authenticates via AD DS
- Federated identity
 - When the user logs into a service, instead of providing credentials to the service provider, the service provider trusts the identity provider to validate the credentials
 - So the user never provides credentials directly to anybody but the identity provider.
- Includes Active Directory Certificate Services (AD CS), Active Directory Lightweight Directory Services (AD LDS), and Active Directory Rights Management Services (AD RMS).

Azure Active Directory vs ADDS

Aspect	Azure AD	Azure AD Domain Services
Device controlled by	Azure AD	Azure AD Domain Services managed domain
Representation in the directory	Device objects in the Azure AD directory.	Computer objects in the AAD-DS managed domain.
Authentication	OAuth/OpenID Connect based protocols	Kerberos, NTLM protocols

Aspect	Azure AD	Azure AD Domain Services
Management	Mobile Device Management (MDM) software like Intune	Group Policy
Networking	Works over the internet	Requires machines to be on the same virtual network as the managed domain.
Extending	Relies on federation to extend scope	Uses trusts between domains for delegated management
💡 Great for	End-user mobile or desktop devices	Server virtual machines deployed in Azure

3.2. Networking

Networking

OSI Layers

1. The physical layer
2. The data-link layer
3. The network layer
4. The transport layer ([Load Balancer](#))
5. The session layer
6. The presentation layer
7. 7 The application layer ([Application Gateway](#))

Azure Traffic Manager

- Used for geo-routing.
- It provides DNS-based routing to redirect end user traffic to globally distributed end points.

Azure Load Balancer

- OSI Layer 4 (TCP & UDP) load balancer that distributes inbound traffic to **backend pools** (resources) according to **rules** and **health probes**.
 - For application-layer processing, see Application Gateway (e.g. SSL off-load)
 - For DNS load balancer (geo-based) see Traffic Manager
- In actual Azure infrastructure there's always a load balancer.
 - No instance is really created, capacity is always available.
- Front-end <=> Load balancer => (through load balancing rules and health probs) Back-end pools (includes VMs)
- Features
 - Instant scaling to applications
 - Reliability via health checks through health probes.
 - Secure: You can add NAT
 - Network Address Translation: remapping one IP to another in VM.
 - Lowers attack surface of the VMs that otherwise could be attacked directly.
 - Port forwarding => A frontend IP to a port of a back-end inside VNet.
 - Agnostic & transparent => Endpoint is only answered by VM (VM TCP handshakes)
- There are quick start ARM templates
 - E.g. 2 VMs internal load balancer.

Load balancer types

Public load balancer

- Internet facing
 - Maps public IP+port of incoming traffic <=> private IP+port.
- E.g. TCP port 80 <=> VMs in web tier subnet in a multi-tiered architecture.
- Can be placed in front of public load balancer to create multi-tier application.

Internal load balancer

- Routes traffic between VMs inside private VNETs or VNETs that use VPN access to Azure.
- Good for:
 - Handling communication within same VNet.
 - Hybrid scenarios: On-prem to VMs in same VNet
 - Multi-tiered applications
 - E.g.: an internal load balancer can receive DB requests that needs to be distributed to back-end SQL servers.

- Critical (line-of-business) applications.
- Frontend IPs and VNets are never directly exposed to internet.
 - Accessed from within Azure or on-prem resources
- Can be used to create multi-tiered hybrid applications.

Load balancer rules

- Determine how traffic is distributed to the backend.
- E.g. you can spread load of incoming web request traffic across multiple web servers.
- Settings: Port, protocol, IP version, back-end port, backend pool, health probe, session persistence, floating IP (enable/disable)
- **Back-end server pool**
 - IP addresses of back-end servers.
 - Back-ends span to two VNets, must be in same VNet.
- Front-end port is called **Listener** and can have SSL certificate.
- Frontend & backend pool are connected with **rules**.
 - Front-end is associated with the back-end VM through **rule definition**.
 - Rules reference to a health probe of target VM.
- You can set **Multiple Frontends**
 - Load balance on multiple ports, multiple IP addresses or both.

ii. **Default rule with no-back-end port reuse**

- Each rule must have a flow with unique IP+port
- Multiple rules can distribute flows to same DIP on different ports.
 - **DIP** = destination IP of VM
- Example:

Rule	Map frontend	To backend pool
1	🎯Frontend1:80	🎯DIP1:80, 🎯DIP2:80
2	🎯Frontend2:80	🎯DIP1:81, 🎯DIP2:81

iii. ***Backend port reuse by using Floating IP**

- If you want to reuse back-end port across multiple rules.

- Good for: e.g. clustering for high availability, network virtual appliances, and exposing multiple TLS endpoints without re-encryption.
- Example:

Rule	Map frontend	To backend pool
1	♥Frontend1:80	♥ DIP1:80, ♥DIP2:80
2	♥♥Frontend2:80	♥ DIP1:80, ♥♥DIP2:80

- **Floating IP** rule allows backend ports to be re-used.
 - It must be enabled.
 - It's a part of DSR (Direct Server Return)
 - Flow topology
 - Outbound part of a flow is always correctly rewritten to flow directly back to the origin.
 - At platform level LB operates this way.
 - IP address mapping scheme
 - By changing destination IP, you can enable port re-use on same VM.

NAT rules

- Must be explicitly attached to a VM (or network interface) to complete the path to the target
- Load balancing rule
 - VM is selected from the back-end address pool or VMs
- 🐾 You would use NAT rule when you have 1 backend server or you know which backend server to get to and load balancing rule when you want to load balance to multiple backend servers.

Session persistence

- 🐾 Provides stickiness to same VM.
- 🐾 A session is a 5-tuple hash of: Source IP, source port, destination IP, destination port (*public port*), protocol (*optional*).

- Settings
 - When adding a rule you can select only client IP or client IP + protocol.
 - Idle timeout: Default: 4 min, same server response via HTTP(S)/TCP sessions but no guarantee that connection is maintained.

Health probe

- Allows resilience and scalability.
- The health probe dynamically adds or removes VMs from the load balancer rotation based on their response to health checks.
- Types:
 - **HTTP**: HTTP 200 => healthy (timeout : 31 seconds)
 - **TCP**: If connection is accepted / refused
 - **Guest probe**: Runs inside VM, not recommended
- Standard SKU sends health probe status as metrics through Azure Monitor.

A high availability ports (HA ports)

- Is a variant of a load balancing rule for internal load balancers.
- Single rule to load-balance all TCP and UDP flows that arrive on all ports of an internal load balancer.
- Non floating: Cannot add more rules.
- Floating: can have same back-end instance or multiple back-ends.

Pricing

- Not mutable, requires re-deploy.
- Basic SKU
 - Free
 - Subset of Standard
 - HA ports are not available
- Standard SKU
 - 💡 New applications should use Standard
 - More flexible & larger backend pools
 - Available only in Standard: Outbound rules (public IP, NAT, HTTPS), SLA
- When back-end pool probes down
 - Standard allows TCP connections to continue
 - Basic terminates all connections.

Azure Application Gateway

- Application Delivery Controller (ADC) as a service
- Web traffic load balancer at "Layer 7 – Application"
 - As opposed to Load Balancer that operates at "Layer 4 – TCP and UDP"
- Application gateway can be configured as
 - Internet-facing gateway
 - Internal-only gateway
 - Combination of both
- You can configure more than one web site on same instance
 - **!** Up to 20 web sites.
 - Each website is directed to its own backend pool of servers.
 - Implementation: 2 backend server pools, 2 listeners (type of multi-site), 2 routing rules.
- Other features
 - **Redirection**
 - One port to other port => enables HTTP => HTTPS and more (e.g. external site)
 - **Session affinity**
 - Keep a user session on the same server
 - **Other features**
 - WebSockets & HTTP/2 traffic, rewrite HTTP headers

Azure Application Gateway components

- Frontend IP Configuration <=> Application Gateway (has WAF and is L7 LB) <=> Listener (through rules) => Backend Server Pool
- **Frontend IP configuration**
 - Traffic-facing port
- **Backend server pool**
 - The list of IP addresses of the backend servers.
 - The IP addresses listed should either belong to the virtual network subnet or should be a public IP/VIP.
 - Every pool has settings like port, protocol, and cookie-based affinity.
- **Listener**
 - Front-end configuration of the gateway.
 - has a front-end port, a protocol (HTTP or HTTPS), and the SSL certificate name (optional).

- Can terminate SSL (SSL offloading), so that traffic flows unencrypted to the backend servers.
 - Unburdens from encryption & decryption overhead.
 - To implement this, upload certificate & set it on listener.
- **Rule**
 - Binds the listener and the backend server pool
 - Defines which backend server pool the traffic should be directed to when it hits a listener.
 - Health probes can be HTTP, HTTPS, default => default webpage, you can configure custom URL.
 - Path based rules allows routing based on paths
 - /video/* => video backend, /picture/* => picture backend
 - Rules are processed in the order they are listed
- **Web application firewall (WAF).**
 - Centralizes security management of web applications = Simpler management.
 - Uses rules from OWASP Core Rule Set:
 - Protection against: SQL Injection (SQLi), Cross Site Scripting (XSS), Local File Inclusion (LFI), Remote File Inclusion (RFI), PHP Code Injection, Java Code Injection, Shellshock, Unix/Windows Shell Injection, Session Fixation, Scripting/Scanner/Bot Detection, Metadata/Error Leakages
 - You can deselect rules or disable the firewall.

Azure Application Gateway sizing

- 3 SKUs

Average back-end page response size	Small	Medium	Large
6KB	7.5 Mbps	13 Mbps	50 Mbps
100KB	35 Mbps	100 Mbps	200 Mbps

- Instance count: Ensures availability: 💡 Min 2 recommended for production

3.2.1. Networking - Hybrid Connections (Site-to-Site VPN & ExpressRoute)

Hybrid Connections

Site-to-Site VPN Connections

- A Site-to-Site (S2S) connection is a connection over IPsec/IKE (IKEv1 or IKEv2) VPN tunnel.
 - Azure-provided name resolution
 - No configuration, azure based address
 - Name resolution that uses your own DNS server
- 📌 In order to do it with on-prem, following steps are followed:
 - i. Create a new custom VNet and gateway subnet
 - Deploy VNet (addresses should be free on-prem)
 - ii. Create a VPN Gateway
 - Deploy **Gateway Subnet** in VNet
 - Deploy **VPN Gateway**
 - Settings:
 - VPN type
 - **Policy based** = static routing
 - **Route based** = dynamic routing
 - Most VPNs are route based.
 - SKU: Affects number of tunnels and the aggregate throughput benchmark.
 - Virtual Networks: Associate VNet with gateway.
 - Each VNet requires its own VPN gateway.
 - You can see IP address in portal after deployment.
 - iii. Add a local site (Local network gateway)
 - +Create a resource => Local network gateway
 - Register IP address and address space of on-prem network.
 - So the routing table can be built by Azure.
 - iv. Configure a VPN device
 - Microsoft lists VPN devices that works well with Azure
 - e.g. Cisco, Juniper, Ubiquiti, and Barracuda Networks.
 - Download & run configuration script for the VPN device.
 - v. Create VPN connection
 - VNet Resource -> Overview -> Connected devices -> VPN Gateway > Connections > +Add
 - Connection type: Select Site-to-site(IPSec)
 - Shared Key: use key from your device or generate key & use in both places

- vi. Verify VPN connection
 - In Connection object you should see "Succeeded"

ExpressRoute

- With ExpressRoute, you can establish connections to Microsoft cloud services, such as Microsoft Azure, Office 365, and CRM Online.
- ExpressRoute connections do not go over the public Internet.
 - More reliable, faster speeds, lower latencies, higher security.
- Good for scenarios like periodic data migration, replication for business continuity, disaster recovery, and other high-availability strategies

ExpressRoute pricing

- You pay for Circuit bandwidth (50 MBPS to 10 50 etc)
- If you choose metered not unlimited => Writing to Azure is free, fetching from Azure costs per GB
- **!** You can have up to 10 virtual networks connections on a standard ExpressRoute circuit, and up to 100 on a premium ExpressRoute circuit.

ExpressRoute connection options

- Connectivity providers can offer one or more connectivity models.
- Features and capabilities are identical.
- **CloudExchange Co-location** (*Layer 2: Data-link, or Layer 3: Network*)
 - If you're in a facility with a cloud exchange, you can use their ethernet exchange.
- **Point-to-point ethernet connection** (*Layer 2: Data-link, or Layer 3: Network*)
- **Any-to-any (IPVPN) Connection** (*typically Layer 3: Network*)
 - You can integrate your WAN with Microsoft cloud.
 - WAN: Wide area network, geographically distributed private telecommunications network that interconnects multiple LANs.
 - Microsoft cloud can be connected to WAN like any other office branch.

ExpressRoute creation

1. On portal select "Resources" => "Create ExpressRoute circuit"
2. Select connectivity provider (Telia etc)
3. Bandwidth => You can always increase, recommended to start low

4. Billing model => Unlimited (flat fee), metered (pay only for data transfer outside Microsoft network, egress)
5. Deployment is done => Microsoft is open to connections
6. Then you enable connection on your side & configure via service key provided in Azure

ExpressRoute monitoring

- Within Log Analytics you have **NPM: Network Performance Monitor**.
- Enables you to monitor
 - throughput, latency, packet loss across various VNets
 - All paths in network in a network map, see latencies across the hubs
- Requires installation of Log Analytics Agent
 - Both on Azure VMs and on-prem in at least one server of each VNet.
- Can be integrated with **SCOM (System Center Operations Management)**
 - It works with Microsoft Windows Server and Unix-based hosts.
 - The agent watches several sources on that computer, including the Windows Event Log

Co-existence of ExpressRoute and Site-to-site

- Why?
 - You can configure a Site-to-Site VPN as a secure failover path for ExpressRoute
 - Site-to-Site VPNs to connect to sites that are not part of your network, but that are connected through ExpressRoute
 - Requires two gateways, one type of VPN, other one type of ExpressRoute.
- ! Limitations
 - Transit routing is not supported.
 - You cannot route (via Azure) between your local network connected via Site-to-Site VPN and your local network connected via ExpressRoute.
 - Transit routing is supported when VNets are peered instead.

3.3. Manage role-based access control (RBAC)

Manage role-based access control (RBAC)

Create a custom role

- 🛠️ Steps (through either PowerShell or Azure CLI)
 - i. Determine the permissions you need
 - Add in Actions and NotActions for data operations use DataActions and NotDataActions.
 - You can see all operations through `az provider operation list`.
 - ii. Create the custom role
 - `az role definition create`
 - You need to have `Microsoft.Authorization/roleDefinitions/write` permissions on all AssignableScopes such as *Owner* or *User Access Administrator*
- ! Limitations
 - Azure supports up to 2000 role assignments per subscription.
 - When you transfer a subscription to a different tenant, all role assignments are permanently deleted. You must re-create your role assignments in the target tenant.

- 🛠️ Example:

```
{
  "Name": "Virtual Machine Operator",
  "Id": "888888888-8888-8888-8888-888888888888",
  "IsCustom": true,
  "Description": "Can monitor and restart virtual machines.",
  "Actions": [
    "Microsoft.Storage/*/read",
    "Microsoft.Network/*/read",
    "Microsoft.Compute/*/read",
    "Microsoft.Compute/virtualMachines/start/action",
    "Microsoft.Compute/virtualMachines/restart/action",
    "Microsoft.Authorization/*/read",
    "Microsoft.Resources/subscriptions/resourceGroups/read",
    "Microsoft.Insights/alertRules/*",
    "Microsoft.Insights/diagnosticSettings/*",
    "Microsoft.Support/*"
  ],
  "NotActions": [],
  "DataActions": [],
  "NotDataActions": [],
  "AssignableScopes": [
    "/subscriptions/{subscriptionId1}",
    "/subscriptions/{subscriptionId2}",
    "/subscriptions/{subscriptionId3}"
  ]
}
```

Configure access to Azure resources by assigning roles

- Through a resource
 - Modify in *Access control (IAM)* in any Azure object.
 - In *Check Access* tab you can check access level of individual user, group, service principal or managed identity.
 - Add a role assignment
 - a. Access control (IAM) -> Add role assignment
 - b. Select **Role** e.g. *Virtual Machine Contributor*.
 - c. In the Select list, select a user, group, service principal, or managed identity.
- For built-in global roles such as *Contributor*:
 - AD -> Users -> Select user -> Directory role -> +Add role
- Assign roles to licenses: Azure Active Directory > Licenses
- Assign roles to devices: Azure Active Directory -> Devices

Conditional access policies

- **When this happens -> then do this**
- You also set Users the users performing an access attempt (**who**).
- Cloud apps: The targets of an access attempt (**what**).
- Controls (**when** this happens)
 - **Grant controls**
 - Support *AND* and *OR* operators.
 - **Compliant device**: Azure AD registered devices, Azure AD joined devices, Hybrid Azure AD joined devices
 - **Hybrid Azure AD joined devices**: Windows desktops, laptops, and enterprise tablets that are joined to an on-premises Active Directory
 - **Approved client app** through Intune app protection policies
 - Enforce acceptance of **Terms of Use**
 - **Custom controls**: Validation through custom APIs.
 - **Session controls**
 - **Use app enforced restrictions**: The device information enables the cloud apps to know whether a connection is initiated from a compliant or domain-joined device.

Azure AD Privileged Identity Management

- Provide **just-in-time** privileged access to Azure AD and Azure resources
- Assign **time-bound access** to resources using start and end dates
- Require **approval** to activate privileged roles
- Enforce **multi-factor authentication** to activate any role
- Use **justification** to understand why users activate
- Get **notifications** when privileged roles are activated
- Conduct **access reviews** to ensure users still need roles
- Download **audit history** for internal or external audit

Azure AD Identity Protection

- **Detecting vulnerabilities and risky accounts**
 - Providing custom recommendations to improve overall security posture by highlighting vulnerabilities
 - Calculating sign-in risk levels
 - Calculating user risk levels
- **Investigating risk events**
 - Sending notifications for risk events
 - Investigating risk events using relevant and contextual information
 - Providing basic workflows to track investigations
 - Providing easy access to remediation actions such as password reset
- **Risk-based conditional access policies**
 - Policy to mitigate risky sign-ins by blocking sign-ins or requiring multi-factor authentication challenges
 - Policy to block or secure risky user accounts
 - Policy to require users to register for multi-factor authentication

Configure management access to Azure

- Securing privileged access requires changes to
 - Processes, administrative practices, and knowledge management
 - Technical components such as host defenses, account protections, and identity management

Recommended roadmap

Stage 1. Critical items (first 24-48 hours)

- Turn on [Azure AD Privileged Identity Management](#) that allows you to
- Identify and categorize accounts that are in highly privileged roles
 - E.g. Global administrator, Privileged role administrator, Exchange administrator, SharePoint administrator
 - Remove any accounts that are no longer needed in those roles.
- Define at least two emergency access accounts
 - E.g. if on-prem AD DS goes down.
 - Highly privileged and are not assigned to specific individuals.
- Turn on multi-factor authentication and register all other highly-privileged single-user non-federated admin accounts

Stage 2. Mitigate common attacks (between 2-4 weeks)

- Conduct an inventory of services, owners, and admins
 - Identify the users who have administrative roles and the services where they can manage.
 - Use Azure AD PIM to find out which users in your organization have admin access to Azure AD, including additional roles beyond those listed in Stage 1.
 - Ensure that your admin accounts have working email addresses attached to them and have registered for Azure MFA or use MFA on-premises.
- Identify Microsoft accounts in administrative roles that need to be switched to work or school accounts
- Ensure separate user accounts and mail forwarding for global administrator accounts
- Ensure the passwords of administrative accounts have recently changed
- Turn on password hash synchronization
- Require multi-factor authentication (MFA) for users in all privileged roles as well as exposed users
- Configure [Identity Protection](#) (= ML to assess risks)
- Monitor **Azure Activity Log**
- Establish incident/emergency response plan owners
- Secure on-premises privileged administrative accounts, if not already done
- Additional steps
 - Complete an inventory of subscriptions using Enterprise Portal
 - Remove Microsoft accounts from admin roles
 - Monitor Azure activity with Azure Activity Log and notifications

- If you have built apps on Azure with on-prem access: Configure Conditional Access policies

Stage 3. Take control of admin activity (between 1-3 months)

- Complete an access review of users in administrator roles
- Continue rollout of stronger authentication for all users
 - E.g. enforce MFA and Windows Hello for high-level managers
- Use dedicated workstations for administration for Azure AD
 - **Privileged Access Workstations (PAWs)**
 - Provides a dedicated operating system for sensitive tasks that is protected from Internet attacks and threat vectors
- Establish integrated monitoring
 - The **Azure Security Center**
 - Provides integrated security monitoring and policy management
 - Across subscriptions
 - Helps detect threats that may otherwise go unnoticed
 - Integrates with a broad ecosystem of security solutions.
- Review [NIST standards](#) and recommendations
- Inventory your privileged accounts within hosted Virtual Machines
 - Grant only the amount of access to users who need to perform specific jobs
- Integrate information protection
 - **Microsoft Cloud App Security (MCAS)** scans & classifies based on policies using **Azure Information Protection** classification labels.
 - E.g. confidentiality levels
- Implement PIM for Azure AD administrator roles
- Use Azure log integrations to send relevant Azure logs
 - To e.g. Azure Sentinel or other SIEM systems

Stage 4: Continue building defenses (after 6 months)

- Review admin roles in Azure AD
- Review users who have administration of Azure AD joined devices
- Review members of built-in Microsoft 365 admin roles
- Validate incident response plan
- Additional steps for organizations managing access to Azure

3.4. Multi-Factor Authentication (MFA)

Multi-Factor Authentication (MFA)

Enable MFA for an Azure tenant

- First you need to install MFA server on Azure (managed).

Enabling options

Enabled by conditional access policy

- E.g. when any user is outside my company network, they're required to sign in with multi-factor authentication
- Cloud only, premium feature of Azure AD
- Flow:
 - Choose verification options:** Azure Active Directory -> Users -> Multi-Factor Authentication -> Service Settings -> Enable verification methods you want: *call to phone, text message to phone, notification through mobile app, verification code from mobile app or hardware token*
 - Create conditional access policy:** Azure Active Directory -> Conditional access -> New policy
 - Select users, groups.
 - Select which cloud apps
 - Select conditions
 - E.g. if you have enabled Azure Identity Protection, you can choose to evaluate sign-in risk as part of the policy.
 - E.g. If you have configured trusted locations or named locations, you can specify to include or exclude those locations from the policy.
 - Under Grant -> Ensure "Grant Access" and "Require multi-factor-authentication" is selected.

Enabled by Azure AD Identity Protection

- Uses the Azure AD Identity Protection risk policy to require two-step verification based only on sign-in risk for all cloud applications
- Cloud only
- Flow: Azure AD -> Identity Protection -> (Configure) User risk policy

Enabled by changing user state

- Requires users to perform two-step verification **every time** they sign in
 - Overrides conditional access policies
- Works with both Azure MFA in the cloud and Azure MFA Server

User states

- All users start out *Disabled*. When you enroll users in Azure MFA, their state changes to *Enabled*. When enabled users sign in and complete the registration process, their state changes to *Enforced*.
- Do not manually change the user state to *Enforced*.
- You can view states for users, and enable:
 - On Portal: Azure Active Directory > Users and groups > All users
 - Powershell script with iteration of users.

MFA settings

- **Block/unblock users:** Azure Active Directory > MFA > Block/unblock users -> Add -> Type e-mail
- **Fraud alerts**
 - Allows users to report fraudulent attempts to access their resources by using the mobile app or through their phone.
 - Specific to on-premises MFA server.
 - Azure Active Directory > MFA > Fraud alert > Set *On* for *Allow users to submit fraud alerts*
 - Configuration options:
 - **Block user when fraud is reported:** If a user reports fraud, their account is blocked for 90 days or until an administrator unblocks their account.
 - **Code to report fraud during initial greeting:** When users receive a phone call to perform two-step verification, user presses a code to report its not user itself.
 - It's 0# by default, you can change it, you should then also change voice greeting with your own recording.
 - **View fraud reports:** Azure Active Directory > Sign-ins
- **One-time bypass**
 - Allows a user to authenticate a single time without performing two-step verification.
 - The bypass is temporary and expires after a specified number of seconds.

- It's good when e.g. in situations where the mobile app or phone is not receiving a notification or phone call.
- **Create:** Azure Active Directory > MFA > One-time bypass -> Enter user e-mail + seconds that the session will last
- **View:** Azure Active Directory > MFA > One-time bypass.
- **Caching rules**
 - You can set a time period to allow authentication attempts after a user is authenticated by using the caching feature
 - It's not intended for Azure AD but on-premises.
 - Set-up: Azure Active Directory > MFA > Caching rules
- **Trusted IPs**
 - Bypasses two-step verification for users who sign in from the company intranet.
 - Premium only feature
 - Options
 - **Managed Azure AD Tenant:** Specific range of IP addresses
 - **Federated:** All Federated Users (only from intranet), Specific range of IP addresses
 - End-user experience outside corpnet: Regardless of whether the Trusted IPs feature is enabled, two-step verification is required
 - Flow:
 - Via conditional access
 - a. Enable named locations by using conditional access
 - Azure Active Directory > Conditional access > Named locations -> New location -> Enter name + IP range + mark as trusted location
 - b. Enable the Trusted IPs feature by using conditional access
 - Azure Active Directory > Conditional access > Named locations -> Configure MFA trusted IPs
 - Via service settings
 - Azure AD -> Users -> Multi-Factor Authentication -> Service settings
 - Two options while enabling trusted IPs:
 - . For requests from federated users originating from my intranet
 - Ensure AD FS adds intranet claim with a rule.
 - a. For requests from a specific range of public IPs
- **Remember Multi-factor Authentication:** Remembers the device.
 - Azure Active Directory > Users and groups > All users -> MFA -> Service settings

3.5.1. Authentication

Authentication

- Act of verifying someone's identity.
- Who are you?

Certificate-based authentication

- Client establish its identity to a server by using a digital certificate.
- 💡 Good as additional security for user authentication.
- In Azure, its handled by AAD.
- You can authenticate to e.g. custom services, Office 365, Skype for Business, API Management, SharePoint..
- Good when multiple front-ends to back-ends architecture is in place.
- Solutions
 - *Traditionally:* Install certificate on each server to establish trust.
 - *In cloud native for hybrid scenarios:*
 - Restrict access to Azure web app.
 - You can use over SSL/TLS
 - You can set API Management to use client certifications.

Active Directory Domain Services (AD DS)

- Introduced with Windows 200 server.
- Hierarchical structure based on X.500
- DNS for locating objects, LDAP for interactions, Kerberos for authentication.
- Create trusts between domains
 - With Organizational Units (OUs) and Group Policy Objects (GPOs) and joining machine

Azure Active Directory

- IDaaS: Identity as a Service
- Identity & access management on Azure with directory services, identity governance and application access management (access control).
- Quickly enables SSO to commercial apps in marketplace and custom apps.
- Can be communicated through
 - Microsoft Graph
 - Azure AD Graph API (! obsolete, use Microsoft Graph)

- Seamless, risk-based data protection controls
 - Backed by ML and in-depth reporting

Managed Identity

- Also known as **Managed Service Identity**, formerly called **Azure Active Directory Managed Service Identity (MSI)**
- A way to keep credentials outside of code.
 - i. Your code calls a local MSI endpoint to get an access token (OAuth2)
 - ii. MSI uses the locally injected credentials to get an access token from Azure AD
 - iii. Your code uses this access token to authenticate to an Azure service
- When you enable MSI for an Azure service:
 - Azure creates a *Service Principal* for the instance of the service in Azure AD
 - Injects the credentials (client ID and certificate) for the *Service Principal* into the instance of the service
- Two types:
 - *System assigned*: Added & managed by azure.
 - Turn on / off: Resource -> Identity blade -> On/Off
 - Enables Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code.
 - RBAC permissions can be controlled in Azure AD
 - *User assigned*: User assigns identities
 - First create Managed Identity
 - It's an ARM object where you can add/remove role assignments.
 - Managed Identities > +Add
 - Resource -> Identity -> User assigned -> +Add
 - You can use this identity to authenticate to services that support Azure AD authentication, without needing credentials in your code.

AD DS vs Azure AD

- Both store directory data and manage communication including logon processes, authentication and directory sources.
- Azure AD is not domain controller in cloud, new way of thinking.
- ADDS is not the solution for cloud because
 - Perimeterless enterprises requires new solution
 - **Perimeter Network (DMZ)** : exposing external services to an untrusted network (e.g. internet).
 - Cloud handles security instead of corporate firewall. to a domain.

- Azure AD differs from AD DS as
 - Has no OUs (organizational units) or GPOs (group policy objects).
 - Uses SAML, WS-Federation or OAuth protocols for authentication.
 - Does not use LDAP but REST API calls to Azure AD or Microsoft Graph API for interaction.
 - Works over HTTP/HTTPS.
 - Azure AD supports [managed identities](#)

Azure AD Connect

- Integrates on-premises directories with Azure AD.
- Three primary components:
- Health monitoring
- Synchronization
 - Creates users, groups and other objects.
 - Ensures match between on-prem <=> cloud
- Active Directory Federation Services (AD FS)
 - Configures hybrid environment with on-prem AD FS.
 - Good for organizations with complex deployments:
 - domain-joins SSO, enforce Azure AD sign-in policy, smart card + third party MFA.
- Features
 - **Filtering:** Limit objects to be synced by filtering by domains, OUs or attributes.
 - **Password hash synchronization:** On-prem AD is the authority so you can use your own password policy.
 - **Password writeback:** Users can reset & change their passwords in cloud and have on-prem policies applied.
 - **Device writeback:** Registered device in Azure AD is written in on-prem AD so it can be used for conditional access.
 - **Prevent accidental deletes:** Limits delete operations (default: 500 per run)
 - **Automatic upgrade:** Updates itself to the latest AD Connect.
- More about AD FS
 - Part of AD DS. Uses it as an identity provider.
 - Consists of two machines that trust each other
 - a. Resource Server (e.g. AD DS):
 - Authenticates & issues a token.
 - b. Federation server (AD FS): identity provider
 - Validates token & issues identity token for other local servers to accept.
 - Flow: App -> Resource Server (AD DS) -> Federation Server (ID provider) (AD FS)

- Provides SSO across organizational boundaries.
- Uses claims-based access-control and implement federated identity.

Legacy authentication methods

Forms-based authentication

- Uses HTML form to send user's credentials to the server.
- Disadvantages:
 - Requires user to interact with HTML web application.
 - Requires measures to prevent cross-site request forgery (CSFR)
 - User credentials are sent in plaintext as part of a request.

Windows-based authentication

- Users log in with Windows credentials using Kerberos or NTLM.
- Credentials are sent in Authorization header.
- Best suited for intranet.
- Disadvantages:
 - Difficult to use in internet without exposing all user dictionary.
 - Can't be used for BYOD (bring your own device)
 - Requires Kerberos or NTLM (Integrated Windows Authentication) support in client browser or device.
 - Client must be joined to Active Directory Domain.
- In hybrid environment, it can be used to manage physical machines and to enable simple Windows-based authentication.

Token-based authentication

Claims-based authentication in .NET

- Evolution
 - *Before*: Federation with SQL DB for user names, passwords, profile data.
 - *Today*: Hard to handle social identity providers when many of them implement storage, tokens & claims
- Identity: a set of claims
 - More expressive than roles (booleans)

- Returned as series of claims by most social providers
- .NET identity implementation
 - Unified identity solution for ASP .NET applications.
 - Implement provider model for logins
 - You can simply add, remove or replace providers (e.g. AD DS, Azure AD, Facebook, Google)
 - Supports claims-based authentication
 - Integrates with most social providers including Azure AD.

App Service authentication and authorization

- It's built-in, no code or SDK is required
 - Runs separately as easyauth.dll
- Claims are injected to .NET and other (e.g. PHP) applications.
- Runs in same sandbox as application code.
- Handles HTTP requests before application.
- Has built-in token store (token repository)
 - You can collect & store & refresh tokens.
 - E.g.:
 - Post to authenticated user's Facebook timeline
 - Read users data from Azure AD Graph API or event from Microsoft Graph.

Multi-factor authentication (MFA)

- One factor (e.g. password) is weak.
- Recommended to have at least two of the following:
 - Knowledge
 - Something only the user knows.
 - *E.g. security questions, passwords, PIN.*
 - Possession
 - Something only the user has.
 - *E.g. corporate badge, mobile device, security token.*
 - Inherence
 - Something that user is.
 - *E.g. fingerprint, face, voice, iris.*
- Implement using Azure AD
 - Two ways to enable
 - Enable for each user:
 - Two-step verification each time users sign in.

- Exceptions: Trusted IP addresses, remembered devices is turned on.
 - **Conditional access policy**
 - Uses the ***Azure AD Identity Protection** risk policy to require two-step verification based only on the sign-in risk for all cloud applications.
- Available verification methods:
 - Call to phone
 - Text Message to phone
 - Through Authenticator App (phone app from Microsoft):
 - Notification through mobile app
 - Verification code from mobile app
- ! Consider using Trusted IPs or named locations as a way to minimize two-step verification prompts.
- Implement using .NET
 - Multi-Factor Authentication SDK for C#, Java, Perl, PHP, Ruby.
 - Each SDK includes a certificate & private key for encryption transactions that are unique to your MFA provider.
 - ! Limitations:
 - APIs have no access to users in Azure AD, you provide them.
 - No enrollment & user management features.

Implementing OAuth2

- [Tutorial](#)
- In Azure AD
 - App registrations -> Register server + client apps with URLs.
 - For client: Settings -> Keys -> Passwords -> Create key with expiration date.
 - Grant permissions to allow the client-app to call the backend-app.
 - AD -> Application registrations -> client-app -> Settings -> Required permissions -> Add -> Select API (backend-app) -> Select permission Access backend-app
- In API Management instance
 - Configure OAuth2.0 server
 - OAuth 2.0 -> Add -> Type Token/Auth URL's (you can find in Endpoints page in AD) -> Add body parameter named resource, type *Application ID* of backend-app -> *Client ID*: use *Application ID* for the client-app -> Client secret: type secret you created in client app -> Copy *redirect_url* -> Create -> Go to *Settings* page of the client-app -> Reply URLs => Paste *redirect_url*

- Protect API with OAuth 2.0
 - APIs -> Select back-end app -> Settings -> Security -> Choose 2.0 -> Select OAuth2.0 server configured earlier -> Save
 - Enable JWT validation for pre-authorizing requests: edit Policy XML add JWT logic.

3.5.2. Authorization

Authorization

- Act of verifying if someone has access to a certain operation/subsystem.
- What can you do?
- Authorization in past
 - Protocols like LDAP or tools like AD DS.
 - Application queried database whenever a user attempted to access an application.
- Today
 - Identity is managed by 3rd parties (Azure AD, Facebook, Google)
 - Information needs to be shared in a standardized way to applications.
- Simplest solution
 - Once users are logged in, ID provider is trusted by application and can share claims.

Claims-based authorization

- To grant or deny access is based on arbitrary logic that uses data available in claims to make the decision
 - Claim
 - Name/value pair that represents what the subject is and not what the subject can do
 - E.g. DateOfBirth = June 8, 1970
- Implementation in .NET
 - Claim-based authorization checks are **declarative**
 - Embedded in code against an action/controller, specifying claims required for current user, and optionally values of claims.
 - Claims requirements are defined in policies.
 - Define policies in `Startup.ConfigureServices`.
 - Require claim(s)


```
services.AddAuthorization(options =>
options.AddPolicy("EmployeeOnly", policy =>
policy.RequireClaim("EmployeeNumber")));
```

- Require claim values

```
services.AddAuthorization(options =>
options.AddPolicy("Founders", policy =>
policy.RequireClaim("EmployeeNumber", "1", "2", "3", "4",
"5")));
```

- Apply policies on action/controller using [Authorize(Policy = "EmployeeOnly")]
 - ⚡ Action overrides controller.

Role-based access control (RBAC) authorization

- An identity can belong to one or more roles.
- Access is granted or denied based on roles.
- ! RBAC is built in ARM so classic deployment cannot use it
- ⚡ Grant users/team least privileges to get their work done.

Setting up RBAC in ASP .NET

- Authorize per role [Authorize(Roles = "HRManager")] Or [Authorize(Roles = "HRManager, Finance")]
- ⚡ Actions overrides controller declarations.
- For only authenticating, you can use [Authorize] and [AllowAnonymous]
- You can declare policies based on roles.
 - In Startup.ConfigureServices
 - Require single role: options.AddPolicy("RequireAdministratorRole", policy => policy.RequireRole("Administrator"))
 - Require multiple roles: options.AddPolicy("ElevatedRights", policy => policy.RequireRole("Administrator", "PowerUser", "BackupAdministrator"));
- On actions and controllers you can then use [Authorize(Policy = "RequireAdministratorRole")]

Role assignment

- Granting access by assigning a **Security Principal**, a **Role** at a **Scope**
- **Security principal**. User, group or service principal.
- **Role** : Built-in or custom role
 - Roles are specific to level, app type (VM, storage)

- **Scope** : Subscription, resource group or resource

Azure built-in roles

- Azure has 70 built-in roles.
- Fundamental roles that apply all resource types:
 - **Owner**: Root, can delegate, can be scoped.
 - **Contributor**: Creates & manages but cannot delegate.
 - **Reader**: Read only access
 - **User access administrator**: Manages user access to Azure resources

3.5.3. Encryption

Encryption

- **Encryption**: Plaintext => Ciphertext, **Decryption**: Ciphertext => Plaintext
- Symmetric encryption
 - Uses symmetric key to encrypt & decrypt
 - The longer the key, the better it is
- Good encryption algorithm: been in use in several years + resisted all attacks.

Encryption at rest

- Encoding (encryption) of data when it is persisted (stored)
- Azure uses symmetric encryption
 - For partitioned data, different keys may be used for each partition.
 - Data encryption keys are often encrypted with asymmetric encryption.
- Flow: *Resource providers* <=> *Data Encryption keys (DEKs)* => *Key encryption keys (KEKs)*
=> *Azure Key Vault (protected by Azure AD)*

Azure Storage

- Enabled by default, cannot be disabled using **Storage Service Encryption (SSE)**
 - Does not affect the performance
 - You can use Encryption Blade to use custom key.
- Some services support customer managed keys & client-side encryption

Azure SQL Database Encryption

- Uses **TDE (Transparent Data Encryption)** on server level as default.
- Encrypts data + log files.
 - Encryption happens in page level
 - Encrypted before writing to disk
 - Decrypted when read into memory
- Encrypts also SQL server + Azure Synapse Analytics (older name: Azure SQL Data Warehouse)
- Does not increase the size of the encrypted database.
- Uses **database encryption key (DEK)**
 - The key is stored in database boot record with a certificate for availability during recovery.
 - It can be symmetric key in master DB server, or asymmetric key protected by EKM (Extensible Key Management)

Azure Cosmos DB encryption

- Database, media attachments, backups are encrypted by default.
 - Primary database is generally stored in SSDs
 - Media attachments & back-ups are stored in Blob storage (generally HDDs)

Always Encrypted

- In Azure SQL Database and SQL server.
- Data is never plaintext inside database.
- Encrypted at rest, during movement between client <=> server, in another words when data is in use.
- Clients decrypts data with a key unknown to DB.
 - Separation between who own the data and who manage the data.
- Requires code changes, driver installed in server, change in connection string.

Azure confidential computing

- Designed when data will never be in plaintext in cloud.
- Hardware vendors help it (e.g. Intel)
 - **Trusted Execution Environment**
 - No one can see the data.

- If code is changed, operations are denied & environment is disabled.
- No code changes required.
- Exposing
 - **Hardware:** Intel CPU's with Intel SGX are available in VMs.
 - **Software:** SGX SDK + 3rd party can be used in VM & compute.
 - **Services:** Many Azure services including Azure SQL.
 - **Frameworks:** Ex: Confidential Consortium Blockchain Framework by Microsoft Research Team.

SSL & TLS

- Security for communications over computer network.
- **Secure Sockets Layer (SSL)**
 - Used by many Azure services.
 - ! SSL (1.0, 2.0, 3.0) are vulnerable and prohibited by IETF.
- **Transport Layer Security (TLS)**
 - ! TLS 1.0 is insecure because of block ciphers (CBC & RC2 CBC) and stream cipher (RC4).
 - 💡 Recommended: Higher than TLS 1.0.
 - Supported by Azure Storage (default: TLS 1.2)

Azure Key Vault

- Secrets store for e.g. password, DB credentials, API key, certificates.
 - Each secret gets URL, vault name must be unique.
- Backed by hardware security modules (HSMs)
- Control & logs access to anything stored in them.
- Can handle requesting & renewing TLS certificates
- Streamlines key management process
 - Dev, test keys and migration to production keys.
- Permissions to keys can be granted and revoked as needed.
- Usage
 - Accessible on Azure Portal
 - **In .NET:** You can use Azure SDK in C# with AzureServiceTokenProvider as authentication callback.
 - **In CLI:**
 - `az keyvault create`
 - `az secret set`
 - `az keyvault secret show`

- **Rest API:**
 - Create key: POST {vaultBaseUrl}/keys/{key-name}/create?api-version=7.0
 - Delete certificate: DELETE {vaultBaseUrl}/certificates/{certificate-name}?api-version=7.0
 - Get secret: GET {vaultBaseUrl}/secrets/{secret-name}/{secret-version}?api-version=7.0
 - Update secret: PATCH {vaultBaseUrl}/secrets/{secret-name}/{secret-version}?api-version=7.0

4.1. Creating Web Applications using PaaS

Creating Web Applications using PaaS

App Service Environments (ASEs)

- Isolated/dedicated environment for securely running App Service apps.
- High scale: Multiple ASEs can be used to scale horizontally
- Isolation and secure network access
 - Isolated running only one customers applications.
 - Always deployed into a VNet.
 - Can have network security groups (NSGs) to control inbound & outbound traffic.
 - Can have web application firewalls (WAFs)
- It's deployed on App Service Plan.
- Can run Web Apps, Docker containers, Mobile Apps, Functions
- Pricing: comes with its own pricing tier "Isolated offering"
- **! Limitations**
 - Everything in-between from
 - Single instance in 100 app service plans
 - 100 instances in a single App Service plan
- Consists of two parts
 - i. **Front-ends:** Responsible for HTTP/HTTPS termination, automatic load balancing.
 - ii. **Workers:** Host customer apps. Three fixed size:
 - One VPU/3.5 GB RAM, Two vCPU/7GB RAM, Four vCPU/14 GB RAM

Web App

- VNET Integration exists only in Standard, Premium, Isolated.
- Create an Azure Service Web App
 - Create resource group => Create app service plan => Create web app => Deploy the app

Web App for Containers

- Choose Docker image
 - or with deployment configuration file for Docker Compose / Kubernetes image from Azure Container Registry, Docker Hub, Private Registry.
- Web App for Containers vs Azure Container Instances
 - **Web Apps for Containers**
 - Deploy web applications on Linux using containers
 - **Azure Container Instances**
 - Easy way to run containers on Azure with a single command without any orchestration.

WebJobs

- Background tasks that enables you to run a program or script in the same context as a web app.
 - E.g. .cmd, .bat, .exe, .ps1, .sh, .php, .py, .js, .jar
- To create you can have CI/CD from GitHub or upload .zip containing your WebJob directly.
- Set "Always On" for reliability.
- 🗓 Scheduled vs continuous

Context	Scheduled	Continuous
Start	When scheduled or triggered manually.	Immediately. Runs in an endless loop.
Multiple instances of WebJobs	On a single instance that's load balanced.	Runs all on instances

Context	Scheduled	Continuous
Remote debugging	Not supported	Supported
Scaling	Not supported	Determines whether the program or script runs on all instances or just one instance. ! Not possible on Free or Shared pricing tiers.

Using Swagger to document an API

- Swagger is set of tools around OpenAPI (JSON standard to define APIs)
- **Swashbuckle**
 - .NET Core package that builds swagger documents from code.
 - Embeds UI for customizing swagger documents.
 - Set up:
 - In `Startup.ConfigureServices`: `services.AddSwaggerGen(c => c.SwaggerDoc("v1", new Info { Title = "My API", Version = "v1" }));`
 - You can here document other information such as *Contact*, *License*, *TermsOfService*, and *Description*.
 - Enable middleware & set-up json UI in `Startup.Configure`:
 - `app.UseSwagger();`
 - `app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "My API V1"));`
 - Swagger is then accessible at `/swagger/v1/swagger.json` and `/swagger`
 - Build swagger from XML documentation:
 - Enable XML documentation file in `.csproj`: `<GenerateDocumentationFile>true</GenerateDocumentationFile>`
 - You'll get warning for undocumented types and members.
 - You can suppress them with inline code or set `<NoWarn>$(NoWarn);1591</NoWarn>` in `.csproj`.

- In `Startup.ConfigureServices` enable XML documents via reflection:


```
c.IncludeXmlComments(Path.Combine(AppContext.BaseDirectory,
    $"{Assembly.GetExecutingAssembly().GetName().Name}.xml"));
```
- You can then just XML document Action results with `<summary>` and `<param name = "id">` tags.
- Declaring models with attributes
 - You can set attributes on view models such as `[Required]` or `[DefaultValue(false)]`
 - On controller you can use `[Produces("application/json")]`
 - On actions you can use `[ProducesResponseType(201)]` and then document with `<response code="201"/>` XML tag.

Azure Logic App

- IPaaS: Integration Platform as a Service
- Automate tasks & integrates apps, data, systems and services by automating tasks and business processes as workflows.
- Workloads e.g.:
 - Move uploaded file from an SFTP or FTP server to Azure Storage.
 - Process and route orders across on-premises systems and cloud services.
 - Monitor tweets for a specific subject, analyze the sentiment, and create alerts or tasks for items that need review.
- 200+ integrations such as Service Bus, Functions, Office 365, Oracle DB etc.
 - Each connector provides triggers and/or actions specifically.
 - **Enterprise Integration Pack**
 - Handles B2B integrations, like BizTalk.
 - Can handle different protocols such as As2, X12, EDIFACT
 - Can transform different formats.
 - Support for encryption & digital signatures.
 - Flow
 - Create an integration account in the portal => Add partners, schemas, artifacts, maps & agreements (reusable artifacts) => Create logic map => Link Logic app to the account => In Logic App, you can use partners and other artifacts from the integration account.
- You can have custom connectors.
 - They're function-based (Azure Functions).
 - Flow: Build your API => Describe your API (Swagger/OpenAPI) and define connector => Use connector => (Optional) Share & Verify your connector
- Applications are defined in a JSON file, they support ARM.

- You can use Azure portal or Visual Studio.
 - Visual Studio
 - You can add apps to source control, publish different versions.
 - Flow: New > Project => Visual C# => Select Resource Group => Modify LogicApp.json => Deploy

4.2. Web Apps (App Service)

Web Apps

- PaaS for web applications.
- Multiple languages and frameworks are supported
 - E.g. ASP. NET, Node.js, Java, PHP and Python
 - Can run any scripts/executables on VMs.
- You can clone app across regions on premium.
- You can create from application templates (e.g. WordPress, Joomla, Drupal).
- Load balancing, auto scaling

App Service Plan

- The infrastructure app services run on.
- Defines compute resources.
- Can be a dedicated VM or shared.
- ! You cannot move it to a different region, but you can clone apps.
- Multiple web apps can run on same app service plan

DevOps

- CI/CD from Azure DevOps, GitHub, Docker Hub and other git sources.
 - Enable on app's menu blade -> App Development -> Deployment Options

Deployment Slots

- **Deployment slot:** Another web-app that's linked to a web-app
 - dev <=> staging <=> production
- Standard => 5, Premium => 20 slots
- All slots have their own names & urls.

- Advantages:
 - Validate app changes in staging before swap (preview before swap)
 - No cold start
 - Ensures all instances are warmed up with no downtime, no request drop
 - If it fails, you can swap back
- Swapping settings
 - Many settings are swapped, some are not.
 - You can choose some Application Settings to be sticky (won't swapped)

Network

- Access on-premises data using Hybrid Connections with VNets.
 - You can use to access e.g. to on-prem database.

App Service Environment

- Clusters servers
 - All the way from front-end load balancers, workers, databases, everything that composes your infrastructure
 - Runs entire configuration in a VNet.
- Configure inbound network traffic rules
- Setup a Web Application Firewall in front of your ASE

Application Settings

- ARR affinity
 - 💡 Turn it off you don't care for states so it scales better.
- 64-bit
 - ! Costs more memory, use it carefully

Always On

- 🛡️ Keeps the app loaded even when there's no traffic
- E.g.
 - When it's on: If VM goes down Azure will try to turn VM on every 10-15 ms.
 - When it's off: VM is brought up when it's needed
- It has no impact on price as you're paying for the entire VM anyways.
- 💡 Keep it

- On: if you want site to be up even when there's no traffic
- Off: If site does not get much traffic (less than every 20 minutes) as it would then free up resources for other sites to use

Backup

- Only Standard or Premium tier.
- You need a storage in same subscription as the app.
- You can back-up databases as well if connection string exists in app settings.
 - works for: SQL Database, Azure Database for MySQL, Azure Database for PostgreSQL, MySQL in-app
- You can automate backup with back-up schedule: How often? When? Retention?
- You can exclude some files.

Snapshots

- For premium or higher, snapshots are taken automatically.
- Advantages over back-ups:
 - No file copy errors due to file locks.
 - No storage size limitation.
 - No configuration required.
- 3 months of snapshots are kept, can only restore for the last 30 days.

Azure Mobile App

- Native & cross-platform apps with e.g. Xamarin, Cordova, Unity.
- Can connect to on-premises.
- Build offline-ready apps with data sync: Syncs when internet is there
- Some special options:
 - **Azure Easy Table**: Light SQL on portal.
 - **Azure Easy APIs**: Code & build directly in Azure portal for Node.js back-end.

App Service Security

Infrastructure and platform security

- Managed by Azure = You trust Azure

- Your app service apps are isolated from both the Internet and from the other customers' Azure resources.
- Communication of secrets (e.g. connection strings) in a resource group does not cross any network boundaries and are always encrypted.
- All communication between App Service and external resources (e.g. REST) are encrypted.
- Other protections:
 - Microsoft Threat Management
 - SQL injection
 - Session hijacking
 - Active: Takes over clients position
 - Passive: Monitors the traffic for password etc.
 - Cross-site-scripting: via click or script tags malicious script is executed.

Azure Security Center

- Natively part of Azure + PaaS services
- A service in the Microsoft Defender Security Center.
- Monitors without any deployment
 - Can protect on-prem by installing **Microsoft Monitoring Agent (MMA)**
- See Network Map of all VNets
- Helps you configure recommended controls
 - E.g. • add firewall • use Azure Active Directory
- Two tiers: • Free • [Windows Defender](#)

Windows defender

- Formerly known as **Azure Security Center Standard tier**
- Allows adaptive application controls and network hardening
- Regulatory compliance dashboard and reports e.g. monitor policies
- [Microsoft Defender for Endpoint](#)
- Threat protection for PaaS services and Azure VMs and non-Azure servers

Microsoft Defender for Endpoint

- Formerly known as **Windows Defender ATP** and then **Microsoft Defender Advanced Threat Protection (ATP)**
- On all VMs out of the box.
- Protects from malware, DDoS, man-in-the-middle attacks and more.

- Man-in-the-middle attacks
 - Reroutes communication between two users through the attackers computers without their knowledge.
 - He can monitor and read the traffic before sending it on the intended recipient.

Application security

- Customers responsibility not Azures.
- AAD integration, certification management, secure communication with different services etc.

Authentication & Authorization

- No code authentication
 - App Service can validate user assertions.
 - Uses JWT tokens
 - ADD => authorization header as bearer token
 - Mobile App Client SDKs
 - Federated identity
 - Identity providers
 - Default: AAD, Microsoft Account, Facebook, Google, Twitter.
 - Can be extended with custom providers.

Client certificate authentication

- Supports SSL certifications.
- Inbound (TLS mutual authentication)
 - Someone who wants to come to your site needs to present a certificate
 - Use API or Resource explorer to set.
- Outbound (using a client certificate from your app)
 - Application talks to another server and that servers requires app to have certification.
 - Set in App settings with WEBSITE__LOAD_CERTIFICATION key.
- Resource explorer = Visualizes APIs and manage rules, you can force require inbound certification there you can also set "expect" one.

Static/Dynamic IP restriction

- Configured in web.config file.

- Static IP restriction: e.g. geo-fencing
- Dynamic IP restriction: e.g. DDoS protection
 - You set request frequency & concurrency rules.

Security scanning

- Via 3rd party "Tinfoil Security"
- Paid & billed outside of Azure
- Managed in Kudo => Extensions
- Use e.g. before you switch slot from production you can see if you introduce any vulnerability.

4.3. Azure Service Fabric

Azure Service Fabric

- Build & manage scalable and reliable microservices that on clusters
 - cluster = a shared pool of machines
- Runtime
 - build distributed, scalable, stateless, and stateful microservices running in containers.
- Management
 - Deploy, monitor, upgrade/patch, and delete deployed applications including containerized services
- You can choose to use ASP.NET Core programming models, or can just deploy and manage containers with Service Fabric container orchestrator.
- Applications are resource-balanced across virtual machines to maximize efficiency.
- Uses cases e.g. real-time data analysis, in-memory computation, parallel transactions, and event processing in your applications

Stateful vs Stateless applications

Stateless services

- Scale with partitioned *storage*
 - Can have dependencies to caches, storages, queues etc., but they're external services.

- As opposed to stateful applications have internal states in them e.g. databases, storages etc.
- Increase reliability with *queues*
- Reduce read latency with *caches*
- Write your own lock managers for *state consistency*

Stateful services

- Maintain state reliably within the service itself, co-located with the code that's using it.
- Application state lives in the compute tier.
- Low latency reads & writes
- Partitions are first class for scale-out
- Built in lock managers based on primary election
- E.g. *Azure Cloud Service worker roles*
- High availability and consistency of the state
 - through simple APIs that provide transactional guarantees backed by replication
 - available by Service Fabric without external store.
- Applications can have their "hot" state and data managed within them for additional performance gains without sacrificing reliability, consistency, or availability
- They're deployed into same Service Fabric cluster with their dependencies.
- Each of these services is independent with regard to scale, reliability, and resource usage, greatly improving agility in development and lifecycle management.
- *Simplify application designs*: They remove the need for the additional queues and caches that have traditionally been required to address the availability and latency requirements of purely stateless applications.

Basic concepts

- **Service type**
 - Service implementation.
 - Defined by class that extends `StatelessService`
 - Has name and a version number
- **Named service instance**
 - An URI using `fabric:/` scheme e.g. `fabric:/MyApp/MyService`
- **Service host**
 - Instances that needs to run inside a host process.
- **Service registration**
 - The service type must be registered with the Service Fabric runtime in a service host

- Allows Service Fabric to create instances of it to run.

Prepare development environments on Windows

1. Install the Service Fabric runtime, SDK, and tools.
2. Enable execution of the Windows PowerShell scripts included in the SDK.

```
Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Force -Scope CurrentUser
```

Creating services

Stateless

Create a stateless project

- New project => Service Fabric Application => Name it HelloWorld
- .NET Core => StatelessService => Name it HelloWorldStateless
- Your solution now contains two projects:
 - HelloWorld
 - Application project
 - Contains services and some PowerShell scripts that help you to deploy your application.
 - HelloWorldStateless
 - Service project
 - Contains stateless service implementation.

Implement stateless entry points

- In service project (HelloWorldStateless.cs) you run any business logic.
- Two entry points:
 - i. Open-ended entry point
 - You begin executing workloads, including long-running compute workloads.
 - `protected override async Task RunAsync(CancellationToken cancellationToken)`
 - `{`
 - `// ...`
 - `}`

- A cancellation token is provided to coordinate when your service instance needs to be closed.
 - Cancellation is requested when:
 - The system moves your service instances for resource balancing.
 - Faults occur in your code.
 - The application or system is upgraded.
 - The underlying hardware experiences an outage.
 - The system will wait for your task to end (by successful completion, cancellation, or fault) before it moves on.
 - It is important to honor the cancellation token, finish any work, and exit `RunAsync()` as quickly as possible when the system requests cancellation.
- ii. A communication entry point
- You can plug in your communication stack of choice, e.g. start receiving requests (e.g. via ASP .NET Core) from users and other services.
- iii. `protected override IEnumerable<ServiceInstanceListener>`
`CreateServiceInstanceListeners()`
- iv. `{`
- v. `// ...`
- `}`

Stateful

Creating a stateful project

- Add -> New Service Fabric Service -> Name it `HelloWorld`
- .NET Core -> Stateful Service and name it `HelloWorldStateful`

Reliable Collections vs Reliable State Manager

- The main difference is the availability of a state provider that can store state reliably.
- State provider implementation called **Reliable Collections**.
 - Lets you create replicated data structures through the **Reliable State Manager**.
 - Operations are transactional = consistent everywhere
 - Stores your data to local disk on each replica.
 - Transactions are atomic

- You may dequeue a work item from a Reliable Queue, perform an operation on it, and save the result in a Reliable Dictionary, all within a single transaction.
- A stateful Reliable Service uses this state provider by default.
- You can store data directly in your service without the need for an external persistent store.

Implement stateful entry points

- A stateful service has the same entry points as a stateless service.
 - In a stateful service, the platform performs additional work on your behalf before it executes `RunAsync()`. This work can include ensuring that the Reliable State Manager and Reliable Collections are ready to use.

- Code example:

```

protected override async Task RunAsync(CancellationToken cancellationToken)
{
    var myDictionary = await
this.StateManager.GetOrAddAsync<IReliableDictionary<string,
long>>("myDictionary");
    while (true)
    {
        cancellationToken.ThrowIfCancellationRequested();
        using (var tx = this.StateManager.CreateTransaction())
        {
            var result = await myDictionary.TryGetValueAsync(tx, "Counter");
            ServiceEventSource.Current.ServiceMessage(this.Context, "Current
Counter Value: {0}", result.HasValue ? result.Value.ToString() : "Value does
not exist.");
            await myDictionary.AddOrUpdateAsync(tx, "Counter", 0, (key, value) =>
++value);
            // If an exception is thrown before calling CommitAsync, the
transaction aborts, all changes are discarded, and nothing is saved to the
secondary replicas.
            await tx.CommitAsync();
        }
        await Task.Delay(TimeSpan.FromSeconds(1), cancellationToken);
    }
}

```

4.4. Using containers and orchestration

Using containers and orchestration

Azure Kubernetes Service (AKS)

- Managed service to deploy Kubernetes cluster.
 - Much of complexity and operational overhead of managing Kubernetes is handled by Azure (e.g.. health monitoring and maintenance)
- Free, pay for the agent nodes within clusters, not for the masters.
- **Deployment & configuration**
 - Through Portal, CLI or ARM
 - Configures all nodes e.g. advanced networking, AAD integration and monitoring
- **Identities**
 - AKS clusters support RBAC.
 - Can be integrated with AAD.
- **Logging**
 - Collects container logs (from stdout and stderr streams) and memory/processor metrics from containers, nodes and controllers.
 - Stored in Log Analytics workspace
 - If RBAC is enabled in Kubernetes, you'll need to grant access to log reader application.
- Cluster nodes
 - **Scaling**: Scale out/in via Azure portal or the Azure CLI.
 - **Upgrades**: Offers multiple Kubernetes versions and upgrading with coordination of nodes to minimize disruption to running applications.
- **HTTP application routing**
 - Configures ingress controller in AKS cluster.
 - As applications are deployed, they're publicly accessible with auto-configured DNS names.
- **GPU enabled nodes**: Supports GPU enabled node pools, provides single/multiple GPU enabled VMs.
- **Development tooling integration**:
 - Tools like *helm*, *Draft*, *Visual Studio Code Kubernetes* works seamlessly.
 - You can run & debug containers directly in AKS.
- **Virtual network integration**: Can be deployed to existing VNet.
 - Every pod is assigned an IP address in the VNet, and can communicate with other pods in the cluster and other nodes in the VNet.
 - Can connect other services in a peered VNet, and on-prem networks over ExpressRoute or site-to-site VPN connections.

- **Private container registry:** Integration with [Azure Container Registry \(ACR\)](#) for Docker images.

Virtual Kubelet

- Connecting Kubernetes to other APIs
- Creates a [Virtual Kubernetes node](#) backed by e.g. Azure, AWS, Alibaba etc.
- Virtual node acts as a normal managed node in the cluster.
- Kubernetes API on top, programmable back.

Deploy an AKS cluster Using CLI

1. Create a resource group: `az group create --name myAKSCluster --location eastus`
2. Create AKS cluster with health monitoring addon: `az aks create --resource-group myAKSCluster --name myAKSCluster --node-count 1 --enable-addons monitoring --generate-ssh-keys`
3. Connect to the cluster
 - i. Download credentials & configure Kubernetes CLI: `az aks get-credentials --resource-group myAKSCluster --name myAKSCluster`
 - ii. Get all nodes: `kubectl get nodes`
 - If you are using Azure Cloud Shell, kubectl is already installed.
 - If you want to install it locally, use the `az aks install-cli` command.

Azure Container Registry (ACR)

- Managed service based on open-source Docker Registry.
- Gives you build automation with build triggers: With such as VSTS or Jenkins. Can be integrated with GIT.
- Use-cases
 - *Scalable orchestration systems:* Can be orchestrated via DC/OS, Docker Swam, Kubernetes.
 - *Azure services:* Azure Kubernetes Service, App Service, Batch, Service Fabric. etc... can build and run applications
- **Azure Container Registry Build (ACR Build)** can build docker images to automate container OS, framework patching pipeline.

Registry

- SKUs: Basic, Standard Premium.
- All support, webhook integration, authentication with AAD, delete functionality.

- You use Azure storages for container images.
- You can use geo-replication in Premium registrations.
- Control access with AAD-backed service principal or provided admin account.

Repository

- A register contains one ore more repositories.
- A repository is group of container images.
- ACR supports multilevel repository namespaces.
- So you can group your repositories.

Image


- Read-only snapshot of a Docker container.
- An Azure container registry can include both Windows and Linux images.
- Control image names for all of container deployments.

Container

- Application and its dependencies wrapped in a filesystem including code, runtime, system tools and libraries.
- Containers running on a single machine share the operating system kernel.
- Containers are fully portable to all major Linux distros, macOS and Windows.

Deploy an image to ACR using Azure CLI

1. Create a resource group: `az group create --name myResourceGroup --location eastus`
2. Create a container registry: `az acr create --resource-group myResourceGroup --name myContainerRegistry007 --sku Basic`
 - Must have unique name.
 - SKUs
 - **Basic:** Have size and usage constraints.
 - **Standard:** Increased storage limits and image throughput.
 - **Premium:** Higher limit on constraints, such as storage & concurrent operations, including enhanced storage capabilities.
 - Higher image throughput capacity.

- Features like geo-replication for managing a single registry across multiple regions, maintaining a network-close registry to each deployment.
3. Log in to ACR: `az acr login --name <acrName>`
 4. Push image to ACR
 5. `docker pull microsoft/aci-helloworld`
 6. `az acr list --resource-group myResourceGroup --query "[].{acrLoginServer:loginServer}" --output table`
 7. `docker tag microsoft/aci-helloworld <acrLoginServer>/aci-helloworld:v1`
`docker push <acrLoginServer>/aci-helloworld:v1`
 8. List container images: `az acr repository list --name <acrName> --output table`
 - Or list tags: `az acr repository show-tags --name <acrName> --repository aci-helloworld --output table`
 9. Deploy image to ACI
 - **Using Azure CLI**
 - Enable admin user in the registry: `az acr update --name <acrName> --admin-enabled true`
 -  In production you should use service principal.
 - Retrieve admin password: `az acr credential show --name <acrName> --query "passwords[0].value"`
 - Create container image with 1 CPU and 1 GB of memory: `az container create --resource-group myResourceGroup --name acr-quickstart --image <acrLoginServer>/aci-helloworld:v1 --cpu 1 --memory 1 --registry-username <acrName> --registry-password <acrPassword> --dns-name-label aci-demo --ports 80`
 - Monitor status of the container: `az container show --resource-group myResourceGroup --name acr-quickstart --query instanceView.state`
 - Retrieve container public IP address: `az container show --resource-group myResourceGroup --name acr-quickstart --query ipAddress.fqdn`
 - **or using docker CLI**
 - Log in to registry: `az acr login --name myregistry`
 - Tag your image: `docker tag nginx myregistry.azurecr.io/samples/nginx`
 - Push your image: `docker push myregistry.azurecr.io/samples/nginx`

10. Docker-push

Azure Container Instances

- Good for simple applications, task automation and build jobs.
- Deprecated, replacement: Azure Kubernetes Service (AKS)

- Features: Full container orchestration including, service discovery across multiple containers, automatic scaling, coordinated application upgrades.
- You deploy an image from a public / private registry

4.5. Serverless Computing (Azure Functions & Logic Apps)

Serverless computing

- Abstraction of servers, infrastructure, and operating systems.
- Main flow: An event triggers code, code gets data, and gives output
- Activity based billing: Only pay when you use resources
 - Billing is based just on resources consumed or the actual time your code is running.
- Serverless applications in Azure:
 - **Compute**
 - [Azure Functions](#)
 - **Storage**
 - [Azure Storage](#)
 - **Database**
 - Azure Cosmos DB
 - **Security and access control**
 - [Azure Active Directory](#)
 - **Cloud messaging**
 - [Event Grid](#)
 - [Service Bus](#)
 - **Workflow orchestration**
 - [Logic Apps](#)
 - **API Management**
 - Azure API Management
 - Azure Function Proxies: single API surface that calls different functions
 - **Analytics**
 - Azure Stream Analytics
 - Event Hubs
 - **Intelligence**
 - Azure Bot Service
 - Cognitive Services

Azure Functions

- FaaS: Functions as a Service
- Built on top of WebJobs SDK.
- No idle capacity charges.
- Execute event-driven code with any programming language.
- Can run locally or in the cloud
 - **Azure Functions Runtime** is free & [open-source](#).
 - Local development requires bindings.
 - Bindings are a way to provide input and output to the function.
 - A function can have multiple input and output bindings.
 - They're defined in a Json file can have different parameters (queueName, tableName) etc. depending on bound target.
- Auto-scaling
 - Scale controller monitors the rate of events and uses heuristics to scales out/in.
 - **!** A single function app will only scale to a maximum of 200 instances.
 - A single instance may process more than message/request at a time though.
- **Pricing**
 - **Consumption Plan**
 - Scale out automatically.
 - Billing is based on number of executions, execution time, and memory used.
 - **App Service Plan**
 - Linux only, you pay for the IaaS.
 - Provides cost predictability and warm start.
- By default, a function will timeout after 5 minutes, and a function can run for a maximum of 10 minutes.
- **Triggers**
 - Can be triggered by Event Hubs, storages, devices from IoT hub, custom sources etc.
 - Or with a scheduled by e.g. webhook, API, data processing.
 - Timer is scheduled with CRON expression
- Bring your own dependencies
 - Supports NuGet, NPM etc., allowing use of preferred libraries.
- Integrated security
 - HTTP-triggered functions can be protected with OAuth providers (AAD, google etc.)
- Code directly in portal or use deployments from GitHub, Visual Studio, etc.

Best practices

- Avoid long running functions
- Use cross function communication
- Prefer stateless functions
- Defensive functions: An exception can happen whenever possible
- For scalability:
 - Share & manage connections
 - Don't mix test and production code in the same function app
 - Use async code but avoid blocking calls
 - Receive messages in batch whenever possible
 - Configure host behavior to better handle concurrency (max outstanding requests, max concurrent requests etc).

Logic Apps

- Cloud & serverless only to orchestrate workflows.
 - Connects serverless functions and APIs.
- Can be integrated with on-prem connectors.
- Integrates data with apps.
- You can manage & manipulate data.
- Many built-in triggers
 - You can have polling triggers, push triggers or recurrence (time-based) triggers.
 - Other built-in triggers include: • HTTP • Request • Azure Functions • Batch • other Azure Logic apps
 - Can be extended.
 - Can be triggered time-based (recurrence scheduling)
- 200+ built-in connectors
 - Managed connectors: • Azure Service Bus • SQL Server • Office 365 Outlook • Azure Blob Storage • SFTP • SharePoint Online • Dynamics 365 CRM Online • FTP • Salesforce • Twitter • Azure Event Hubs • Azure Event Grid, ...
 - On-prem connectors: • BizTalk Server • File System • IBM DB2 • IBM Informix • MySQL • Oracle DB • PostgreSQL • SharePoint Server • SQL Server • Teradata
 - For additional cost: • Enterprise Connectors (SAP, IBM MQ..) • Integrations account connectors (B2B messages via special protocols, XML handling)
 - Each connector has specific actions
 - E.g. twitter has: • Post a tweet • Get followers • Get following
- Built logic with

- Error handlers
- Conditionals: for each, condition (if), scope, switch, terminate, until.
- Manage variables with expressions (such as div, concat etc)
 - Data operations: compose, create html/csv, filter array, join, parse json, select
 - Date Time: add to time, convert time zone, current time, get future time, get past time, subtract from time
 - Variables: append to array, append to string, decrement, increment, initialize, set

5.1. Asynchronous Messaging

Asynchronous Messaging

High-performance computing (HPC)

- Low-latency Remote Direct Memory Access (RDMA) networking
 - Low-latency network connection between the processing running on two servers or virtual machines in Azure.
 - It copies data from the network adapter directly to memory and avoids wasting CPU cycles.
 - For developers, it's seen like single machine with shared memory
- Usage: Installed on **head node** (must be Windows)
 - Server can manage **compute nodes** (can be Linux or Windows)
- How it works:
 - Uses Intel MPI library.
 - Some VMs (A8,A9 or ends with 'r') have network interface for RDMA
- Use-cases:
 - Burst to Azure from on-prem
 - molecular modeling and computational fluid dynamics
- In Azure:
 - H-series VMs

Azure Batch

- PaaS for HPC
- Manages VMs, computes nodes with job scheduling (uses internal queue), autoscale, data persistence.

- Batch API lets you wrap existing (on-prem or cloud) compute node so it runs as a service on computer node pool that Batch manages.
- Good for intrinsically parallel (sometimes called **embarrassingly parallel** or **perfectly parallel**) workloads
 - Little to no effort to make application parallel because there's little to no dependencies between them
 - Image processing
- Integrates with **Azure Data Factory** (pipeline solution in Azure)
- Usage
 - Create Batch Account
 - In Batch account, create *pools* of computer nodes (VMs)
 - In Batch account, create *job* that runs basic *tasks* on pool
 - A task is a CMD script

Queues

- Split application into smaller modules with asynchronous messaging
 - Easy-to-manage in long term: Modules can be swapped/modified/updated without having to update the entire application.
- See also [Azure Queues](#)

Message queues

- Queue: Buffer that supports send & receive operations.
- Supports following cases:
 - Sender can send message to the queue
 - Receiver receives message from the queue
 - Message is removed.
 - Receiver examines (peeks at) the next available message
 - Message is not removed in the queue.
 - Many queues supports also returning the length of messages or if there's message left in queue to avoid blocking the receiver.
 - Many support transactions, leasing and visibility on top of it.
 - Leasing = concurrency lock
 - Message is unavailable to other receives while one receiver handles it.

Concurrent queue consumers

- Does not block the business logic while requests are processed.
- Too many requests => consumer service is overflowed
 - Solution:
 - They need to be scaled out horizontally.
 - Only once instance should get the message
 - Load Balancer is needed to avoid one receiver becomes bottleneck
- In Azure: Azure Storage queues, Azure Service Bus queues.

Webhooks

- Polling => overallocating resource to check data.
- Use user-defined HTTP callbacks
- In azure functions you use it with Azure Functions.
 - function.json defines when the function is triggered.

◦ Example for webhooks

```

◦ {
◦     "disabled":false,
◦     "bindings":[
◦         {
◦             "authLevel":"function",
◦             "name":"req",
◦             "type":"httpTrigger",
◦             "direction":"in",
◦             "methods":[
◦                 "post"
◦             ]
◦         },
◦         {
◦             "name":"$return",
◦             "type":"http",
◦             "direction":"out"
◦         }
◦     ]
◦ }

```

◦ Example for GitHub trigger

```

◦ {
◦     "bindings":[
◦         {
◦             "type":"httpTrigger",
◦             "direction":"in",
◦             "webHookType":"github",

```

```

○      "name": "req"
○      },
○      {
○          "type": "http",
○          "direction": "out",
○          "name": "res"
○      }
○  ],
○  "disabled": false
○  }

```

5.1.1. Azure Queues (Storage Queues & Event Grid & Service Bus & Event Hubs & IoT Hub)

Azure Queues

- Queues using pull model: • [Storage queues](#) • [Service bus](#) • [Event Hubs](#) • [Azure IoT Hub](#)
- Queues using push model: • [Event grid](#)
- ! Queues using pull model can be integrated with [Event grid](#) to push the events.

Storage Queues

Implement Azure storage

Send message

```

CloudStorageAccount storageAccount =
CloudStorageAccount.Parse(CloudConfigurationManager.GetSetting("StorageConnectionStri
ng"));
CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient();
CloudQueue queue = queueClient.GetQueueReference("myqueue");
queue.CreateIfNotExists();
CloudQueueMessage message = new CloudQueueMessage("Hello, World");
queue.AddMessage(message);

```

Receive message

- Peek: CloudQueueMessage peekedMessage = queue.PeekMessage();
- Read & dequeue
 - Get message (message becomes invisible for 30 seconds): CloudQueueMessage retrievedMessage = queue.GetMessage();
 - Delete message to dequeue: retrievedMessage.DeleteMessage(retrievedMessage);

Service Bus


- PaaS for web applications.
- Managed messaging infrastructure across private & public cloud.
- Uses pull model, can be integrated with [Event Grid](#) to have push model
- Extends [Storage queues](#) with middlewares, publish/subscribe messaging, load balancing, FIFO.
- Can be used to connect on-prem with cloud or cloud to cloud.

Service Bus Concepts

- Communication mechanisms
 - **Queues** (simple queues)
 - Each queue acts as broker (intermediary) that stores messages until sent.
 - One directional broker: Message is sent to single recipients.
 - **Topics** (publish-and-subscribe)
 - One-directional broker: single topic can have multiple subscriptions.
 - Can use filters
 - **Relays** (connects with direct communication)
 - Bidirectional communications
 - No storage or broker: just passes messages to the destination.
- **Namespaces**
 - Application scoped container for all messaging components.
 - Can have multiple queues & topics.
 - Can have different communication mechanisms
- **Messages**
 - Decouples applications.
 - Enables 1:n relationships with topics & subscriptions
 - Message sessions enables message ordering or deferral.
 - Decouples applications: improves scalability and reliability.
- **Queues**
 - Offers FIFO with ordering + timestamps or more competing consumers.
 - Messages are delivered in pull mode, held in redundant storage.

Implement Azure Service Bus

- Using C#

- Send batch to queue
- ```
QueueClient queueClient = new QueueClient(ServiceBusConnectionString,
QueueName);
string messageBody = $"First Message";
Message message = new Message(Encoding.UTF8.GetBytes(messageBody));
await queueClient.SendAsync(message);
var messages = new List<Message>();
for (int i = 0; i < 10; i++)
{
 var message = new Message(Encoding.UTF8.GetBytes($"Message
{i:00}"));
 messages.Add(message);
}
await queueClient.SendBatchAsync(messages);
```
- Handle a message
  - In client SDK, you have handler callback you call either CompleteAsync or AbandonAsync (you receive the message again)
- Create namespaces where pub/subs will meet
  - Provides a service + security boundary.
  - Can be integrated with **Azure Relay**
    - Integrates on-premises communication easily.
    - Less intrusive than VPN.
    - Traditional one-way, request/response, and peer-to-peer communication
    - Event distribution at internet-scope to enable publish/subscribe scenarios
    - Bi-directional and unbuffered socket communication across network boundaries.
  - The name provides a unique identifier for the object. For example, sbces12345.servicebus.windows.net
- Create a queue
  - You can select: • Message time to live • Lock duration • Duplicate detection (duplicates won't be accepted) • Dead lettering (hold messages can't be delivered in another queue) • Sessions (guarantees FIFO) • Partitioning.
-  Receiving an event
  - ReceivesAndDelete
    - Simplest, ok if system can tolerate if a message is missing if it can't be handled
  - PeekLock
    - CompleteAsync => Message is handled & deleted
    - AbandonAsync => Re-queued

## Publish/Subscribe

- **Temporal decoupling**
  - Producers (senders) and consumers (receivers) do not have to be sending and receiving messages at the same time, because messages are stored durably in the queue.
- **Load leveling**
  - Producers consumers send & receive messages at different rates.
  - You don't have to pay for a system that is underutilized part of the time.
- **Loose coupling**
  - Resilience because messages are durable until they reach the worker.
- **Load balancing**
  - Bring on more workers as the queue increases.
- **Topics and subscriptions**
  - One to many (receivers)
  - Messages can be filtered in topics
- Other features
  - Auto-forwarding
  - Batching
  - Scheduled delivery (delayed processing)
  - Message deferral

## Azure Service Bus Pricing

- **Premium**
  - Fixed size
  - Predictable performance
  - Up-down scaling
  - Messages up to 1 MB
- **Standard**
  - Elastic
  - Auto-scaling
  - Message size up to 256 KB

## Service Bus Monitoring

- Metrics
  - **Request metrics** counts the number of requests.
  - **Message metrics** counts the messages (active, incoming, outgoing etc)



- **Connection metrics** active/opened/closed messages
  - **Resource usage metrics** in Premium: CPU/memory size usage per namespace.
- Diagnostics Logs
  - E.g.
    - ActivityId • EventName • resourceId • SubscriptionId • EventTimeString • EventProperties • Status • Caller • category (= operationalLogs)

## Service Buses vs Storage Queues

- Service buses are built on top of storage queues and are more advanced:
  - Clients can use AMQP (ISO standard for queueing)
  - FIFO is guaranteed with sessions.
  - And other features such as batch send, automatic dead lettering, message auto-forwarding, message groups, duplicate detection, sessions, transactions, duplicate detection, durable publish/subscribe.
- In storage queues is REST only, FIFO is not guaranteed, lease/lock is on message level (while it's on queue level in service buses).

## Event Grid

- iPaaS: Integration Platform as a Service
- Fully managed, http-based event routing service for events.
- It connects and integrates all Azure services.
  - You can create events in Events blade of resources (e.g. blob storage)
- Eliminate polling because it's expensive.
  - Long polling: Server waits until the data is available instead of empty directly.
    - it is very expensive in terms of CPU, memory and bandwidth
- Concepts
  - Event Publishers (sources) => *(through topics)* Event Grid => *(through subscriptions)* Event Handlers
  - **Topic:** Endpoint where the source sends events.
  - **Event Subscription:** what you're interested in receiving
    - Can be filtered
    - Can have expiration date.
- Batching is supported (array of events) and recommended.
- Use case example:
  - Ops automation: New VM created or SQL DB spun up => check whether configurations are compliant, tag, file work items etc.
- You can use custom events

- Get easy to use UI
  - Native integrated event handlers
  - Uniform consumption of events
- Uses pub/sub model.
  - Can send events to multiple recipients
- Pricing:
  - Pay by number of operations
  - Published events + delivery attempts - monthly free grant (100k) = total operations x \$0.60

## Event Hubs

- Big data streaming platform and event ingestion service.
  - **Event ingestion**
    - "Front door" for an event pipeline
    - Sits between publishers and consumers.
- Seamless integration with data + analytics services inside/outside Azure.
- Key components
  - **Event Producers**
    - Can publish using HTTPS, AMQP 1.0 or Apache Kafka.
  - **Partitions**
    - Views of consumers (state + position + offsets)
  - **Throughput units**
    - Pre-purchased units to control capacity of EventHubs
  - **Event receivers**
    - Connect via AMQP 1.0.
- Can be integrated **Azure Stream Analytics**
  - Azure Stream Analytics: PaaS for parallel Complex Event Processing (CEP) pipelines for BI.

## Azure IoT Hub

- Central message hub for bidirectional communication between IoT application and the devices it manages.
- Use cases include device to cloud telemetry, file upload from devices, request-reply.
- You can run Azure services (Azure Functions, Azure Stream Analytics, Azure Machine Learning, ..), or your own code on-prem on devices with remote cloud monitoring & management
- **EventHubs vs Azure IoT Hub**
  - Azure IoT Hub leverages Event Hubs for its telemetry flow path

- Both support the ingestion of data with low latency and high reliability

## 5.2. Configure Message-based integration architecture

### Configure Message-based integration architecture

- See also [Azure Queues](#)

## Configure to send emails

- Use SendGrid
  - Flexible API, scalable, real-time analytics & sent messages, transactional.
  - C# has SDKs.

## Configure an event and publish model

- **Stipulation:** Requirement that applications will handle high volume and velocity.
  - **Stipulate:** make an agreement or to demand
  - Applications in serial manner is difficult to meet the stipulation
  - Solution event driven architecture.
- An event-driven architecture consists of **event producers** that generate a stream of events and event **consumers** that listen for the events.
- Decoupling: producers from consumers, consumers from each other.
- Common implementations
  - **Simple event processing**
    - Event immediately triggers an action.
    - E.g. Azure Functions with Azure Service Bus trigger.
  - **Complex event processing**
    - Looks at series of events to look for patterns.
    - E.g. Azure Stream Analytics or Apache Storm.
  - **Event stream processing**
    - Streaming platform as pipeline to ingest event and feed them to processors.
    - Ex. Azure IoT Hub or Apache Kafka.

## Azure Event Grid

- Built-in supports for incoming events from Azure services, supports also custom application & third party events via webhooks/custom topics.
- You can filter & route events to different points, or use multicasting to send events to multiple end-points.
- Concepts:
  - **Event sources** sends **Topics**.
  - **Event handlers** listens to **Topics** via **Event Subscriptions**
- Subscribe to Blob Storage events to send to an endpoint
  - Supported only in Blob Storage or General Purpose v2
  - Using Azure CLI
    - `az group create --name DemoGroup --location eastus`
    - `az storage account create --name demostor --location eastus --resource-group DemoGroup --sku Standard_LRS --kind BlobStorage --access-tier Hot`
    - `az eventgrid event-subscription create --resource-id $storageid -name contosostoragesub --endpoint https://contoso.com/api/update`
- Security and authentication
  - Three types:
    - Webhook event delivery
      - ValidationCode handshake (programmatic)
        - Application gets validation code & echoes back.
      - ValidationURL handshake (manual)
        - Send GET request to validation URL.
    - Event subscriptions
      - Requires Microsoft.EventGrid/EventSubscriptions/Write permission on the resource.
    - Custom topic publishing
      - SAS (recommended) or key authentication: include the resource, an expiration time, and a signature

## Configure the Azure Relay service

- Reveals on-prem to cloud for different communications (request/response, P2P, publish/subscribe)
  - Especially bidirectional via webhooks.
- It only uses outbound port from on-prem for extra security.
- Flow
  - Client => Azure Relay service => Gateway node (*each has its own relay/input*)
    - It's listening request (bidirectional)?
      - Creates a new relay
    - It's a request to a specific relay
      - Gateway forwards to gateway node that's own the relay.
      - Relay owning gateway sends **rendezvous request** to client.
        - Rendezvous: temporary channel to exchange messages.
- In Node.js
  - ws package: WebSocket protocol client library
  - hyco-ws package: Extends ws where Azure Relay is built.
  - Use hycows.Server instead of ws.Server, mostly contract compatible
- Applications can authenticate to Azure Relay using Shared Access Signature (SAS) authentication
  - You set permissions as authorization rules in a Relay namespace.

## Create & configure a Notification hub

- Azure Notification Hub provides server to client mobile app communication through push notifications.
- From any back-end (cloud or on-prem) to any platform (iOS, Android, Windows..)
- Delivered through **Platform Notification Systems (PNSs)** to Platform specific infrastructures
  - Provides handle to push notifications
  - No common interface, different for IOS, Android, Windows.
- Multi-platform, scaled abstraction of PNSs, no handles, send notifications there and it routes further to users/interest groups.
- cases: location based coupons to segments, breaking news to all, codes for MFA.
- **Flow** : Client app contacts PNS to retrieve push handle, it stores the handle and uses to push notifications.
- SDKS for IOS, Xamarin, C# exists.

## Configure with Microsoft Graph

- Unified API for Azure AD, Office 365, Windows 10 (*activities and devices*), Microsoft Enterprise Mobility (*Microsoft Identity Manager, Microsoft Intune, Microsoft Advanced Threat Analytics, and Microsoft Defender for Identity*)
- Uses relationships such as `memberOf`

### 5.3. Developing for asynchronous processes

## Development for asynchronous processes

### Asynchronous Processing

- Multiple CPU cores enables multiple threads.
- In past, it was complex with locks, took years to master.
- Today, it's simplified with ex. TPL (Task Parallel Library).
- Task Parallel Library (TPL)
  - Extends `System.Threading` to simplify parallelism and concurrency.
  - Dynamical scaling of degree of concurrency for maximum efficiency.
  - Handles partitioning of work, scheduling threads in thread pool, cancellation, state management and more.
  - A task holds metadata about the delegate: whether delegate has started/completed executing and it's resulting value.
  - Use `async-await` to refactor TPL code.

### Serverless computing

- Useful for "gluing" together disparate systems (e.g. integration solutions)
- They can all define input, actions, conditions, and output.
- You can run each of them on a schedule or via a trigger.

### Azure Functions

- FaaS: Functions as a Service
- Support many languages such as C#, F#, Node.js, Java or PHP.
- Excellent for: data processing, integrating systems working with IoT, simple APIs/micro-services.
- Integrates with Azure and 3rd party services
  - Integrations can trigger your function or serve as input/output.

- You can edit & compile & test (with request manipulations) & save in Portal.
- One function app can have many functions.
- In .NET
  - Provided by Azure WebJobs SDK.
  - json specifies input/outputs.
  - ```
{
  "disabled": false,
  "bindings": [ {
    "type": "queueTrigger",
    "direction": "in",
    "name": "message",
    "queueName": "announcementqueue",
    "connection": "StorageConnectionString"
  } ]
}
```
 - Required fields: direction, name, type
 - C# script binds to "in" property as a method parameter:
 - ```
public static void Run(string message, System.Text.StringBuilder log)
{
 log.AppendLine($"New message: {message}");
}
```

## Logic Apps

- Build & schedule & automate processes as workflows.
- Scenarios: B2B, on-prem or cloud only or both, data/system integration, enterprise application integration (EAI).
- You can integrate with on-premises using an **On-premises Data Gateway**.
  - i. Download and install gateway
  - ii. Create Azure resource for gateway (**on-premises data gateway**)
  - iii. Connect to on-premises data
    - Add a connector that supports on-premises connections, for example, SQL Server.
    - Select *Connect via on-premises data gateway*.
- Build visually in Logic Apps designer, available in Azure portal & Visual Studio. You can also use PowerShell & ARM templates for select tasks.

- Describe in a json template:
- ```
{
  "$schema": "https://schema.management.azure.com/schemas/2016-06-01/Microsoft.Logic.json",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "uri": { "type": "string" }
  },
  "triggers": {
    "request": { "type": "request", "kind": "http" }
  },
  "actions": {
    "readData": {
      "type": "Http",
      "inputs": { "method": "GET", "uri": "@parameters('uri')" }
    }
  },
  "outputs": {}
}
```

Implement interfaces for storage or data access

- Blocking the calling thread during I/O
 - Thread enters a wait state which other operations can't use it.
 - Anti-pattern as it reduces performance & bad for vertical scalability.
- Asynchronous code
 - Do not wait for I/O, transition long-running work other CPU cores.
 - Allows servers to handle more requests & UIs to be more responsive.
- Use asynchronous interfaces for all data tier code, pair those with interfaces with asynchronous versions.
 - E.g. if you have Entity Framework in controller, you'll need to rewrite code if you use another mapper or database in future => Use interfaces with async implementation.

5.4. Developing for autoscaling

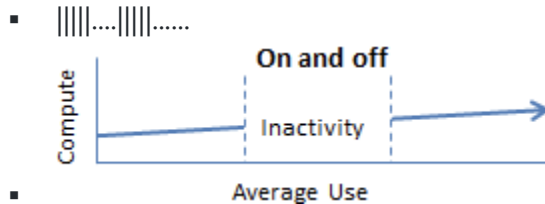
Developing for autoscaling

- Applications workloads are unpredictable
 - Overestimate => Pay for unnecessary compute resources
 - Underestimate => Poor user experience
 - Ideally => Use extra instance only when it's needed and shut down when it's not.

Workload Patterns

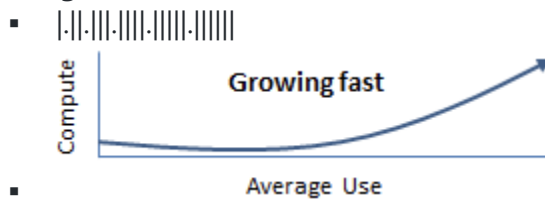
- 🏰 Four common computing patterns you'll see for web applications in cloud

- **On and Off**



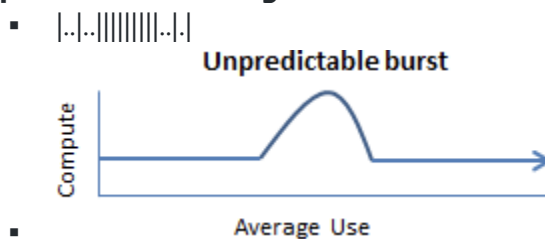
- E.g.: batch processing.

- **Growing fast**

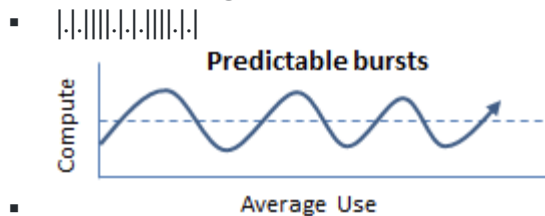


- Often growing start-ups.

- **Unpredictable bursting**



- **Predictable bursting**



- E.g. during black friday for a e-commerce site.

- Distribute applications across multiple instances to provide redundancy + performance.
 - A load balancer is needed to distribute.

Auto scale

- Primary advantage of the cloud is **elastic scaling**.
 - Ability to use as much capacity as you need, scale out if load increases, scale in when the extra capacity is not needed.
- Supported in many Azure Services

- IaaS: Azure Virtual Machine Scale Sets (identical VMs in same set)
 - PaaS: Azure App Service
 - Or event database services such as Cosmos DB
- Auto-scale metrics
 - Supported in all pricing plans of App Service.
 - Autoscale can be triggered based on metrics or at scheduled date and time.
 - Metrics are aggregated over all instances of the plan
 - E.g. CpuPercentage, MemoryPercentage, BytesReceived, BytesSent, HttpQueueLength, DiskQueueLength (read+writes queued on storage)
- ! Basic plan does not include AutoScaling

5.5. Implement code that addresses a transient state

Implement code that addresses a transient state

Durable functions

- An extension of Azure Functions that lets you write stateful functions
 - The extension manages state, checkpoints, and restarts for you.
- Logic
 - You get starter object injected in JS & C# (DurableOrchestrationClient)
 - If starter => existing instance (instanceid) exists
 - return HttpStatusCode.Conflict
 - else
 - starter => start new instance
 - starter => create response from instanceid
 - return response from starter

5.6. Implement code that addresses singleton application instances

Implement code that addresses singleton application instances

- An application that communicates with elements running in the cloud has to be sensitive to the transient faults that can occur in this environment.
 - Faults e.g. momentary loss of network connectivity to components and services, the temporary unavailability of a service, or timeouts that occur when a service is busy.
- These faults are self-correcting and if action is done after delay, it's likely to be successful.

- E.g. `ConnectionClosed`, `TimeOut`, `RequestCanceled`
- Strategies
 - **Cancel** : Report exception & cancel operation. E.g. invalid credentials.
 - **Retry** : If specific fault reported is unusual or rare, E.g. network packet becoming corrupted.
 - **Retry after delay** : Fault caused by e.g.. busy/connectivity failures. Try after short period of time.
 - For more common transient failures, period between retries should be chosen to spread requests from multiple instances of the application as evenly as possible
 - Reduces chance of being overloaded.
 - Too many service retry => longer to recover
 - If service fails again, wait & make another attempt, if necessary, increase delays between retry attempts until maximum is reached.
 - Delay can be increased incrementally or exponentially depending on the type of failure & probability that it'll be corrected during this time.
- Many SDKs implement retry policies, where some parameters can be set: maximum number of retries, amount of time between retry,
- An application should log the details of faults & failing operations.
- Scaling out can lower frequency of faults caused by being overloaded etc.
 - Partition the database & spread the load across multiple servers.
- In code
 - Try catch for the exception
 - Set delay (`Delay = TimeSpan.FromSeconds(5)`) and wait for the delay (`Task.Delay`)
 - Log the exception
 - throw if retry count is maximum

5.7. Querying Azure Resources

Querying Azure Resources

Using Azure CLI

- Azure CLI uses `-query` argument to execute a JMESPath query
 - JMESPath => JSON query language.
 - `-query` argument is supported by all commands in the Azure CLI.

- Return type
 - JSON Array, no order guarantee.

- Projection

- E.g. select in LINQ

```
az vm list --query '[].{name:name
image:storageProfile.imageReference.offer}'
```

- Filtering

- E.g. where in LINQ

```
az vm list --query "[?starts\_with(storageProfile.imageReference.offer,
'WindowsServer')]"
```

- Combine project + filter

```
az vm list --query "[?starts\_with(storageProfile.imageReference.offer,
'Ubuntu')].{name:name, id:vmId}"
```

Using fluent Azure SDK

- Better option if you intend to write code to find connection information for a specific application instance.
- Flow:
 - Connect
 - You need azure.auth file (JSON file describing, secret, key url's etc)
 - You can create like this: `az ad sp create-for-rbac --sdk-auth > azure.auth`
 - Then Azure `azure = Azure.Authenticate("azure.auth").WithDefaultSubscription();`
 - See VMs


```
var vms = await azure.VirtualMachines.ListAsync();
foreach(var vm in vms)
{
    Console.WriteLine(vm.Name);
}
```
 - Gather virtual machine metadata to determine the IP address

- `INetworkInterface targetNic = targetVm.GetPrimaryNetworkInterface();`
- `INicIPConfiguration targetIpConfig = targetNic.PrimaryIPConfiguration;`
- `IPublicIPAddress targetIpAddress =`
`targetIpConfig.GetPublicIPAddress();`
`Console.WriteLine($"IP Address:\t{targetIpAddress.IPAddress}");`

5.8. Develop database solutions

Develop database solutions

Develop solutions that use Cosmos DB storage

- **Create, read, update, and delete data by using appropriate APIs**
 - Use async, use interfaces!
 - **SQL API:** Standard API.
 - `DocumentClient.CreateDocumentAsync`, `DocumentClient.ReadDocumentAsync`, `DocumentClient.ReadDocumentFeedAsync` (*read all documents*), `DocumentClient.CreateDocumentQuery`, `DocumentClient.ReplaceDocumentAsync`, `DocumentClient.UpsertDocumentAsync`, `DocumentClient.DeleteDocumentAsync`.
 - **Azure Cosmos DB's API** for MongoDB
 - **Gremlin API:** It's used to store and operate on graph data. Gremlin API supports modeling Graph data and provides APIs to traverse through the graph data.
 - **Cassandra API:** You can switch from using Apache Cassandra to using Azure Cosmos DB 's Cassandra API, by just changing a connection string.
 - **Table API:** Azure Table storage can migrate to Azure Cosmos DB by using the Table API with no code changes and take advantage of premium capabilities.
- **Implement partitioning schemes**
 - Cosmos DB uses hash-based partitioning to spread logical partitions across physical partitions. The partition key value of an item is hashed by Cosmos DB, and the hashed result determines the physical partition.
 - **! Limitations**
 - A single logical partition is allowed an upper limit of 10 GB of storage.
 - Partitioned containers are configured with minimum throughput of 400 RU/s. Requests to the same partition key can't exceed the throughput allocated to a partition.
 - It's important to pick a partition key that doesn't result in "hot spots" within your application.
 - Syntactic partition keys

- It's a best practice to have a partition key with many distinct values, such as hundreds or thousands.
- Concatenate multiple properties of an item
 - E.g.
 - {
 - "deviceId": "abc-123",
 - "date": 2018,
 - "partitionKey": "abc-123-2018"
 - }
- Use a partition key with a random suffix
 - Distribute the workload more evenly is to append a random number at the end of the partition key value.
 - You can perform parallel write operations across partitions.
 - E.g. you might choose a random number between 1 and 400 and concatenate it as a suffix to the date.
 - like 2018-08-09.1, 2018-08-09.2, and so on, through 2018-08-09.400
 - This method results in better parallelism and overall higher throughput
 - The randomizing strategy can greatly improve write throughput, but it's difficult to read a specific item
- Use a partition key with precalculated suffixes
 - Easier to read than randomizing.
 - E.g. you have ID, you create partitions key with date + hash of ID.
 - The writes are evenly spread across the partition key values, and across the partitions.
 - You can easily read a particular item and date because you can calculate the partition key value for a specific ID.
 - The benefit of this method is that you can avoid creating a single hot partition key.
 - **A hot partition key** is the partition key that takes all the workload.
- You can provision throughput for a CosmosDb database or container during creation e.g. OfferThroughput = 100000
- **Set the appropriate consistency level for operations**
 - Make the fundamental tradeoff between the read consistency vs. availability, latency, and throughput.

- Strong consistency model (or linearization)
 - Adds a steep price of higher latency (in steady state) and reduced availability (during failures).
 - Easy to program applications
- Eventual consistency
 - Higher availability and better performance
 - Hard to program applications.
- Consistency levels in Cosmos Db (from strongest to weakest)
 - **Strong, Bounded staleness, Session, Consistent prefix, Eventual**
 - **Strong:** Always guaranteed to read the latest committed write.
 - **Bounded staleness**
 - Staleness can be configured in two ways:
 - The number of versions (K) of the item
 - The time interval (t) by which the reads might lag behind the writes
 - 💡 Recommended for high consistency
 - **Session**
 - Scoped to client session.
 - honors the *consistent-prefix* (assuming a single "writer" session), *monotonic reads*, *monotonic writes*, *read-your-writes*, and *write-follows-reads* guarantees
 - Monotonic reads: If a process reads the value of a data item x, any successive read operation on x by that process will always return that same value or a more recent value.
 - Monotonic writes: A write operation by a process on a data item X is completed before any successive write operation on X by the same process.
 - 💡 Recommended option for most scenarios.
 - **Consistent prefix**
 - Updates that are returned contain some prefix of all the updates, with no gaps. Consistent prefix guarantees that reads never see out-of-order writes.
 - **Eventual:** There's no ordering guarantee for reads. In the absence of any further writes, the replicas eventually converge.
 - **Consistent prefix**, and **eventual consistency** provide about two times the read throughput when compared with strong and bounded staleness.
 - For a given type of write operation, the write throughput for request units is identical for all consistency levels.

- You can configure the default consistency level on your Azure Cosmos account at any time

Develop solutions that use a relational database

Provision and configure relational databases

- Create single database
 - i. Create a resource -> Databases -> SQL Database
 - Type database name, select subscription, resource group
 - Select source: Blank, Sample, Back-up
 - ii. Server -> Create new
 - Type server name, admin login, password, location
 - iii. Select pricing tier
 - Standard/Basic/Premium (sets minimum DTU)
 - Select DTUs & maximum storage
 - iv. You can query in SQL Database -> Query Editor
- Create managed instance
 - Azure SQL Database Managed Instance is a fully managed SQL Server Database Engine Instance hosted in Azure cloud. This is the best PaaS option for migrating your SQL Server database to the cloud.
 - The managed instance deployment option provides high compatibility with on-premises SQL Server Database Engine.
 - Cross-database query support is one of the main reasons to use managed instance over Azure SQL Database.
 - **!** Elastic Transactions / The Microsoft Distributed Transaction Coordinator is not supported.
 - **Flow:** Create a resource -> Managed Instance -> Azure SQL Managed Instance
 - Set e.g. collation, VNet
 - Features:
 - Availability: Always-on, backup
 - Security: Auditing, certificates, credentials through Azure Key Vault or Shared Access Signature
 - Set cryptographic providers
 - Configuration:
 - Collation
 - Defines a collation of a database or table column, or a collation cast operation when applied to character string expression

- Sets the alphabetical order based on culter, e.g. Latin1_General_CS_AS_KS_WS, or Traditional_Spanish_ci_ai
- SQL Server Agent: Server options, e.g. replication options such as snapshot, transaction-log reader, notifications.
- Functionalities
 - Bulk insert / openrowset
 - Distributed transactions: ! Neither MSDTC (The Microsoft Distributed Transaction Coordinator) nor Elastic Transactions are currently supported in managed instances.
 - Service broker, stored procedures/functions/triggers
- Behavior changes: Returns through e.g. ServerProperty('InstanceName')

Configure elastic pools for Azure SQL Database

- SQL Database elastic pools
 - Simple, cost-effective solution for managing and scaling multiple databases that have varying and unpredictable usage demands.
 - The databases are on a single Azure SQL Database server.
 - Share a set number of resources at a set price
- Problem & solution
 - **Problem:**
 - A common application pattern is to provision a single database for each customer.
 - But different customers often have varying and unpredictable usage patterns, and it is difficult to predict the resource requirements of each individual database user.
 - You over-provision or under-provision
 - **Solution**
 - Elastic pools solve this problem by ensuring that databases get the performance resources they need when they need it. They provide a simple resource allocation mechanism within a predictable budget.
- eDTUs are shared between many databases.
 - Costs 1.5x more than DTU's but pool eDTUs can be shared by many databases and fewer total eDTUs are needed.
- **Choose the correct pool size**
 - Determine
 - Maximum resources utilized by all databases in the pool.
 - eDTUs: $\text{MAX}(\text{Total number of DBs} \times \text{average DTU utilization per DB})$
 - Maximum storage bytes utilized by all databases in the pool.

- Sum of storage needed per DB
- **Elastic jobs**
 - Management tasks are simplified by running scripts in elastic jobs.
- **Flow:**
 - i. **Create pool:** SQL elastic pool -> +Add -> Create pool on new server or existing SQL server.
 - ii. In pool -> **Configure pool**
 - Select a service tier, add databases to the pool, and configure the resource limits for the pool and its databases.
 - DTU-based resource limits
 - Max eDTUs per database, Min eDTUs per database, Max storage per database
 - vCore-based resource limits
 - Max vCores per database, Min vCores per database, Max storage per database
- Rescaling
 - When rescaling pool, database connections are briefly dropped
 - The duration to rescale pool can depend on the total amount of storage space used by all databases in the pool

Elastic transactions

- Allows you to write queries against multiple databases.
- If both SQL databases support elastic transactions, you can create server communication link.
 - New-AzureRmSqlServerCommunicationLink: Create a new communication relationship between two SQL Database servers in Azure SQL Database. The relationship is symmetric which means both servers can initiate transactions with the other server.
- In C#
 - **Multi-database applications**
 - The TransactionScope class establishes an ambient transaction.
 - Ambient transaction = transaction that lives in the current thread.
 - All connections opened within the TransactionScope participate in the transaction. If different databases participate, the transaction is automatically elevated to a distributed transaction.

- Ex:

```

▪ using (var scope = new TransactionScope())
▪ {
▪     using (var conn1 = new SqlConnection(connStrDb1))
▪     {
▪         conn1.Open();
▪         SqlCommand cmd1 = conn1.CreateCommand();
▪         cmd1.CommandText = string.Format("insert into T1
values(1)");
▪         cmd1.ExecuteNonQuery();
▪     }
▪
▪     using (var conn2 = new SqlConnection(connStrDb2))
▪     {
▪         conn2.Open();
▪         var cmd2 = conn2.CreateCommand();
▪         cmd2.CommandText = string.Format("insert into T2
values(2)");
▪         cmd2.ExecuteNonQuery();
▪     }
▪
▪     scope.Complete();
▪ }

```

- **Sharded database applications**

- Use `OpenConnectionForKey` method of the elastic database client library to open connections for a scaled out data tier.

- C#:

```

▪ using (var scope = new TransactionScope())
▪ {
▪     using (var conn1 =
shardmap.OpenConnectionForKey(tenantId1, credentialsStr))
▪     {
▪         conn1.Open();
▪         SqlCommand cmd1 = conn1.CreateCommand();
▪         cmd1.CommandText = string.Format("insert into T1
values(1)");
▪         cmd1.ExecuteNonQuery();
▪     }
▪
▪     using (var conn2 =
shardmap.OpenConnectionForKey(tenantId2, credentialsStr))
▪     {
▪         conn2.Open();
▪         var cmd2 = conn2.CreateCommand();
▪         cmd2.CommandText = string.Format("insert into T1
values(2)");
▪         cmd2.ExecuteNonQuery();
▪     }
▪ }

```

```

    ▪
    ▪     scope.Complete();
    }

```

- Monitoring
 - Use Dynamic Management Views (DMVs) in SQL DB.
 - E.g. sys.dm_tran_active_transactions, sys.dm_tran_database_transactions, sys.dm_tran_locks.

Create, read, update, and delete data tables by using code

- First ensure you can reach the database: Database -> Set server firewall -> Add client IP
- You create/delete table, populate table or update/delete and select data with TSQL
- Example using C# + ADO .NET:
 - ```
string query = "UPDATE [guitarBrands] SET type = @type, name = @name, image = @image WHERE id = @id";
```
  - ```
using (SqlConnection connection = new
SqlConnection(ConfigurationManager.ConnectionStrings["brandsConnection"].ToString()))
using (SqlCommand command = new SqlCommand(query, connection)
{
    try
    {
        // open the connection, execute, etc
        List<SqlParameter> p = new List<SqlParameter>();
        p.Add(new SqlParameter("@type", newType.Text));
        p.Add(new SqlParameter("@name", newName.Text));
        p.Add(new SqlParameter("@image", newImage.Text));
        p.Add(new SqlParameter("@id", id));
        connection.Open();

        /* Submit query
        command.ExecuteNonQuery(); // Non query does not return results
        */

        /* Read
        using (SqlDataReader reader = command.ExecuteReader())
        {
            while (reader.Read())
```

```

•      {
•          Console.WriteLine("{0} , {1} , {2} , {3} , {4}",
•              reader.GetGuid(0),
•              reader.GetString(1),
•              reader.GetInt32(2),
•              (reader.IsDBNull(3)) ? "NULL" : reader.GetString(3),
•              (reader.IsDBNull(4)) ? "NULL" : reader.GetString(4));
•      }
•  } */
•  }
•  catch
•  {
•      // log and handle exception(s)
•  }
•  }

```

6.1. Design and Connectivity Patterns

Design and Connectivity Patterns

Partitioning workloads

- Modularize application to functional units.
- Each module
 - Handles portion of application's overall functionality
 - Represents set of related concerns.
- Why?
 - Easier to design both current & future iterations of your application.
 - Modules can also be tested & distributed and otherwise verified in isolation.

Load balancing

- Application traffic or load is distributed among various endpoints by using algorithms.
- Allows
 - Multiple instances of the website can be created
 - They can behave in a predictable manner
 - Flexibility to grow or shrink the number of instances in application without changing the expected behavior
- Load balancing strategy considerations
 - Physical vs Virtual Load balancers

- Use Virtual Load Balancers (hosted in VMs) if company requires a very specific configuration.
- Load balancing algorithm
 - round robin => Selects next instance for each request based on a predetermined order that includes all of the instances.
 - random choice
- Configurations
 - Affinity/stickiness: If subsequent requests from the same client machine should be routed to the same service instance.
 - Required when application has state.

Transient fault handling

- Leads to more resilient applications.
- Implemented in .NET libraries (Entity Framework, Azure SDK etc)
- Transient errors=> occur due to temporary interruptions in the service or to excess latency.
- Many are self-healing and can be resolved with retry policy
- Retry policy
 - Retry when a temporary failure occurs.
 - A break in the circuit => abort retries if it's a serious issue.

Queues

- Provides a degree of consistency regardless of the behavior of the modules.
- Direct method invocation
 - Connection is severed on transient errors
 - Use 3rd party queue to persist the requests beyond a temporary failure.
 - Allows you to audit failing requests independently.

Retry pattern

- Cloud applications must be sensitive to transient faults.
 - E.g. loss of network connectivity, the temporary unavailability of a service, timeouts that arise when a service is busy.
- They're typically self-correcting, if the action that triggered a fault is repeated after a suitable delay, it's likely to be successful.
 - DB with too many concurrent requests can have throttling (fails until workload is eased). Fixes itself after some delay.

- Solution: Retry for temporarily fails.
 - Remote service => retry after short wait.
 - Fails again => Limit attempts to avoid brute forcing retry again until maximum tries are reached.
 - to spread requests from multiple instances of the application as evenly as possible.

Competing consumers pattern

- Sudden large number of requests may cause unpredictable workload.
- Single consumer => risk of being flooded, or messaging system being overloaded.
- Solution: asynchronous messaging with variable quantities of message producers and consumers
 - Business logic in the application is not blocked while the requests are being processed.
 - Handle fluctuating workloads => system can run multiple instances of the consumer service.

Cache-aside pattern

- Problem: Cached data consistency
 - A strategy is needed to ensure that the data is up-to-date & handle situations where the data in cache has become stale.
- Solution: read-through and write-through caching
 - Cache-aside => Effectively loads data into the cache on demand if it's not already available in the cache.
 - Not in cache? Fetch & add it to cache, modifications on cache => write to data store.

Sharding pattern

- Problem: hosting large volumes of data in a traditional single-instance store
- Some limitations
 - Storage space: Upgrading disks is not easy.
 - Computing resources: It's not possible to always increase more
 - Network bandwidth: Network traffic might exceed
 - Geography: Reduce latency of data access for different across regions.
- Scaling up can postpone affects but only temporary solution
- Solution: partitioning data horizontally across many nodes

- Divide data store into horizontal partitions, or shards.
- Shard: Same schema but distinct subset of data.
- Sharding can be in data access code, or storage system with transparent sharding
 - Abstracting physical location => High level of control over which shard contain which data.
 - Easier to migrate between shard without touching application logic.
 - Tradeoff => Additional data access overhead to determine the location of each data item as it's retrieved
- For optimal performance & scalability
 - Split data in a way that's appropriate for the types of queries the application performs.
 - Sharding schema will exactly match requirements of every query.
 - E.g.:
 - In multi-tenant system => You lookup with tenant id + e.g. tenant's name, Tenant's name = sharding key

6.2. Hybrid Networking

Hybrid Networking

Site-to-site connectivity (Site-to-site VPN)

- Between your on-premises site <=> VNet in Azure via IPsec tunnel.
- Resources on local network can communicate with resources on Azure VNet
 - No need for separate connection for each client computer in local network.
- Requires VPN device.
- E.g.:
 - IT Pros and Developer in-office have their own gateway and connect to Azure.
 - Q&A offshore team has its own gateway and connect to Azure

Point-to-site connectivity (Point-to-site VPN)

- Configured on each client computer that you want to connect to the VNet in Azure.
- No need for VPN device
 - Instead you use VPN client you install on each client computer.
 - Requires manually starting connection from client, can have auto reset.

Combining site-to-site and point-to-site connectivity

- Q&A offshore team connects via VPN gateway (site-to-site VPN)
- Developers & IT Pros at office connects via VPN gateway (site-to-site VPN)
- Developers working from home connect via direct VPN (point-to-site VPN)

Combining ExpressRoute and site-to-site connectivity

- Reasons
 - Multiple branch offices, it's costly to purchase peering for every location.
 - Multiple networks within the enterprise
 - Connect one to Azure using Express route for higher-risk traffic.
 - For lower-risk traffic, use site-to-site VPN
 - Use site-to-site VPN as a failover link if ExpressRoute connection fails.

Virtual network to virtual network connectivity (VNET to VNET)

- Utilizes Azure VPN gateways to connect VNets in Azure over IPSec/IKE tunnels.
- E.g.: you have following topology (topology=nodes connect to other network via links)
 - IT-pros/developers in office has VPN-to-VPN to *Azure East Asia*
 - Offshore QA team has VPN-to-VPN to *Azure West US*
 - You set VNet-to-VNet between *Azure East Asia* and *Azure West US*
 - Then both team can access *Azure East Asia* and *Azure West US*

Connecting across cloud providers

- For failover, backup or migration between providers.
- Amazon Web Services (AWS) =>
 - Create EC2 VM with Openswan (VPN software)
 - Create gateway on the Azure VNet side using static routing.
 - Use gateway IP from Azure to configure Openswan for tunnel connection

6.3. Storing in cloud

Storing in cloud

Durability of data

- A transaction is set of operations.
 - Seek to achieve some or all ACID properties.
 - **Atomic**
 - A transaction is executed only once; all work completes or none does.
 - Why?
 - Operations in a transaction often share common intent or depend on each other.
 - Performing only subset => intent can be missed.
 - **Consistent**
 - A transaction preserves the consistency of data.
 - Performed on consistent state and leads to consistent state.
 - Typically, developers are responsible for maintaining consistency.
 - **Isolated**
 - Concurrent transactions behave as if each were the only transaction running in the system.
 - Some applications reduce isolation level for better throughput
 - High isolation => limits number of concurrent transactions
 - **Durable**
 - A transaction must be recoverable.
 - It must be persisted if e.g. computer crashes.
 - Special logging solves this.
 - In relational database systems (RDBMS) it's a single unit of work.
 - All-or-none => If it fails, DB is rolled back, all modification are erased.

Caching

- Caching aims to improve performance & scalability of a system.
- It's done by temporarily copying frequently accessed data to a fast storage, close to application.
- Most effective when
 - Same data is repeatedly read.
 - Original data store =>

- Relatively static
- Slow compared to cache's speed
- Subject to significant level of contention
 - Contention in DB systems =>
 - multiple processes or instances competing for access to the same index or data block at the same time
 - It's far away & network latency cause access to be slow.
- Distributed applications typically implement either or both when caching data:
 - **Private cache** : Locally held on computer that's running application.
 - **In-memory store**: Accessed by single process.
 - Quick & effective, size is typically constrained to host machine.
 - **Local file system**
 - Slower than in-memory, but faster than retrieving across network.
 - Each application holds its own copy of the data.
 - Problem:
 - Snapshot of the original data at a point of past.
 - Different application instance can hold different versions.
 - **Shared cache** : Common source which multiple processes/machines can access.
 - All instances see same view of data as opposed to in-memory.
 - It's highly scalable
 - Cache services uses cluster of servers and software for distribution.
 - Easy to scale by adding to / removing from a cluster.
 - Disadvantages:
 - Slower to access => Held locally to each application instance.
 - Implementing separate cache service => increases complexity.
- Caching considerations
 - When?
 - The more data you have, the larger number of users that need to access this data => minimum load on the original data store.
 - If original data store is unavailable, cache can be used.
 - How to cache data effectively?
 - Determine the post appropriate data to cache
 - Cache it at the appropriate time.
 - Add data to the cache on demand when it's retrieved first time.
 - Populate in advance
 - **Seeding**: when the application start.
 - Not good for large cache as it can cause sudden high load.
 - Manage data expiration.
 - Cached data becomes stale after a while.
 - Expire caches so they're removed, and retrieved on next read.

- Set a default policy, many cache services you can set period for individual objects while storing them programmatically.
- **Redis Cache**
 - Recommended by Azure, replaces Azure Cache (deprecated).
 - NoSQL key-value database.
 - Unique: Allows complex data structure for its keys.
 - SKUs: Basic (single node), Standard (2 nodes + SLA)

Measuring throughput

- Normalized units
 - Relative performance guarantees by cloud vendors.
 - your application uses 20 units, 40 unit will give you appr. double performance.
- DTUs – Database throughput units (Azure SQL Database)
 - Based on compute, storage and IO.
 - DTUs for single databases, eDTUs for elastic pools.
 - Fixed per pricing plans, e.g.: Basic = 5 DTU, Standard 2 = 50 DTU
- RUs – Request unit processing per second (Azure Cosmos DB)
 - Each operation incurs a request charge, which is expressed in Rus.
 - Single request unit (normalized) => 1 read of 1 KB document.
 - Create, replace, delete consumes more processing = more request units.

Structure of data

- Polyglot persistence
 - Solutions that uses mix of data store technologies.

Structured data stores

- Most vendors use SQL.
- Have RDMS (relational database management system)
 - Conforms to be ACID.
 - Supports schema-on-write
 - You define data structure, all read+write use same schema.
- Hard to scale out.
- E.g. Azure SQL Database, Azure Database for MySQL, Azure Database for PostgreSQL

Unstructured / semi-structured data stores

- Doesn't use tabular schema of rows & columns.
- Can store as key/value pairs, JSON documents, or as a graph (edges + vertices)
- Have no relational model.
- Graph databases => • Cosmos DB • Gremlin API
 - Optimized for exploring weighted relationships between entities.
 - Stores edges (entities) and nodes (relationship between nodes).
- Document databases => Azure Cosmos DB
- NoSQL => Most systems supports SQL compatible queries, but non-SQL DBs.
- Column family: HBase in HDInsights
 - Key-value pair, where key is mapped to a value that's a set of column.
- Massively parallel & distributed solutions for ingesting, storing, and analyzing data
 - Azure Synapse Analytics (older name: SQL Data Warehouse)
 - Azure Data Lake
 - Time series data stores => Time Series Insights
 - Optimized for queries over time-based sequences of data, indexed by datetime.
- Others: Object storage => Blob storage, Shared files => File storage

1.1. Security - Responsibilities

Responsibilities

- Microsoft gives you a secure foundation & tooling to control your environment but customers have the responsibility of their subscription governance, data, identities, and how to protect those.
- The cloud presents a spectrum of responsibilities based on what types of services and/or features a customer may be consuming.
 - This is unlike more traditional on-premises information systems where most, if not all, security is implemented by the same owner.
 - In IaaS, customer owns more control than in PaaS or SaaS.
- The subscription is associated with a Microsoft account or organizational account.

1.2. Security - Azure data centers

Azure data centers

- Azure data centers are secured by using different technical isolations.
- Based on following components:
 - Azure Fabric Controller
 - Virtualization
 - Logical Separations

Azure Fabric Controller (FC)

- Kernel of the Azure platform, managing resources as needed.
- Provisions, stores, delivers, monitors and commands the VMs and physical servers that make up the Azure customer environment and infrastructure.
 - Deploys & manages health of compute services.
 - Manages data center infrastructure (hardware & software), recovers from failures
 - Drives infrastructure updates.

Virtualization

- The **Host OS** is a configuration-hardened version of Windows Server.
- The **Hypervisor** is Hyper-V from Windows Server 2012 R2, which has been battle-tested and proven in enterprise environments worldwide.
 - Two types of a hypervisor:
 - Type 1 Hypervisor (*e.g. VMware, HyperV*) runs the OS.
 - Type 2 Hypervisor (*e.g. VMware Workstation, VirtualBox*) runs on OS.
- The **Guest VM OS** can be either Windows Server, several distributions of Linux, or an OS image supplied by the customer (much be supported Operating Systems, or starting from the Azure Marketplace images).

Logical separations

- Segregates each customer's data & application from that of others.
- **Storage isolation**
 - **Storage Access Key (SAK)**: Data is accessible only through claims-based Identity Management & access control with a Storage Access Key.
 - **Shared Access Signature (SAS)**
 - Recommended as it does not reveal account key and is more granular & restricted access.

- Can be reset via the Microsoft Azure Portal or the Storage Management API.
 - Storage blocks are hashed by the hypervisor to separate accounts.
- **SQL isolation:** SQL Azure isolates separate account databases.
- **Network isolation:** VM switch at the host level blocks inter-tenant communication.

1.3. Security - Azure Key Vault.

Azure Key Vault

- Helps safeguard cryptographic keys and secrets used by cloud applications and services.
- You can encrypt keys and secrets (such as authentication keys, storage account keys, data encryption keys, .PFX files, and passwords) by using keys that are protected by hardware security modules (HSMs).
- You can import or generate keys in HSMs. If you choose to do this, Microsoft processes your keys in FIPS 140-2 Level 2 validated HSMs (hardware and firmware).
 - HSM = Hardware security module
- Streamlines the key process and enables you to maintain control of keys that access and encrypt your data.
 - Developers can create keys for development and testing, and then seamlessly migrate them to production keys.
 - Security administrators can grant (and revoke) permission to keys, as needed.
- Administration
 - It can be created and managed by an organization's administrator who manages other Azure services for an organization.
 - E.g.:
 - Administrator would sign in with an Azure subscription, create a vault for the organization in which to store keys, and then be responsible for operational tasks, such as:
 - Create or import a key or secret.
 - Revoke or delete a key or secret.
 - Authorize users or applications to access the key vault, so they can then manage or use its keys and secrets.
 - Configure key usage (for example, sign or encrypt).
 - Monitor key usage.
 - This administrator would then provide developers with URIs to call from their applications, and provide their security administrator with key usage logging information.

1.4. Security - Azure Active Directory (Azure AD)

Azure Active Directory (Azure AD)

- PaaS
- Multi-tenant, cloud-based directory and identity management service.
- Combines core directory services, advanced identity governance, and application access management.
- Allows IT admins to give SSO too many services including Office 365, Salesforce.com, DropBox and Concur.
- Full suite of identity management capabilities including multi-factor authentication, device registration, self-service password management, self-service group management, privileged account management, role-based access control, application usage monitoring, auditing and security monitoring and alerting.

Single Sign-On

- Also called identity federation
- Hybrid-based directory integration scenario of Azure Active Directory that you can implement when you want to simplify your user's ability to seamlessly access cloud services with their existing Active Directory corporate credentials
- A **Secure Token Service (STS)** enables identity federation,
- Extends centralized authentication, authorization and SSO to e.g. Web applications & services, including perimeter networks, partner networks, and the cloud.
- Configuring a STS is creating a federated trust between your on-premises STS and the federated domain you've specified in your Azure AD tenant.

Azure AD Authentication Strategies

Azure AD Connect

- Always a required sync tool.
- Supports synchronization from multiple Azure AD Forests/Domains, into a single Azure Active Directory environment.
- Enables cloud way of authenticating:
 - i. Password Hash Sync
 - ii. Federation (ADFS)

- iii. Azure AD Passthrough Authentication Agent
- The underlying database can be a SQL Server Express, or a full SQL Server 2008 R2 or newer database instance.
- Can be installed on dedicated VMs, or directly on ADDS Domain Controllers
- AD Connect requires an **AD Connect Service Account**
 - Reads/write information from the Azure AD Tenant, as well as requiring an on-premises account in Active Directory, Enterprise Admin level rights, to read/write information back in the on-premises Active Directory.
 - Allows for a two-way sync, e.g. password resets (optional – requires P1), account deletions and other strategies for connecting.

Azure AD Connect Health

- Monitoring of on-premises identity infrastructure.
- It enables you reliable connection to Office 365 and Microsoft Online Services.
- Rich usage metrics: Use the Azure AD Connect Health portal to view alerts, performance monitoring, usage analytics, and other information
- Get alerted on all critical ADFS system issues
- Easy to deploy and manage


Azure AD B2B vs B2C

- Use B2C e.g. when users need to authenticate via facebook to your application.
 - Azure AD B2C implements OpenID Connect, which supports many different providers, including Twitter, Google, and many others that support the standard.
 - Azure AD B2C protects from denial-of-service and password attacks against your applications and includes user interface customizations to easily integrate into your existing applications.
- Use B2B e.g. when inviting consultant companies.


Category	Azure AD B2B	Azure AD B2C
<i>Intended for</i>	To be able to authenticate users from a partner organization, regardless of identity provider.	Inviting customers of your mobile and web apps, whether individuals, institutional or organizational customers into your Azure AD.

Category	Azure AD B2B	Azure AD B2C
<i>Identities supported</i>	Employees with work or school accounts, partners with work or school accounts, or any email address. <i>(Soon to support)</i> direct federation.	Consumer users with local application accounts (any email address or user name) or any supported social identity with direct federation.
<i>Which directory the partner users are in</i>	Partner users from the external organization are managed in the same directory as employees, but annotated specially. They can be managed the same way as employees, can be added to the same groups, and so on.	In the application directory. Managed separately from the organization's employee and partner directory (if any).
<i>Single sign-on (SSO)</i>	Single sign-on to all Azure AD-connected apps is supported. E.g. you can provide access to Office 365 or on-premises apps, and to other SaaS apps such as Salesforce or Workday.	Single sign-on to customer owned apps within the Azure AD B2C tenants is supported. ! SSO to Office 365 or to other Microsoft and non-Microsoft SaaS apps is not supported.
<i>Partner lifecycle</i>	Managed by the host/inviting organization.	Self-serve or managed by the application.
<i>Security policy and compliance</i>	Managed by the host/inviting organization.	Managed by the application.
<i>Branding</i>	Host/inviting organization's brand is used.	Managed by application. Typically tends to be product branded, with the organization fading into the background.


Multi-Factor Authentication

- Requires more than one verification method.
- E.g. any two or more of the following verification methods:
 - Something you know (*typically a password*)
 - Something you have (*a trusted device that is not easily duplicated, like a phone*)
 - Something you are (*biometrics*)
- **Azure Multi-Factor Authentication (MFA)**
 - Microsoft's two-step verification solution
 - Verification methods includes phone call, text message, or mobile app verification.
 - You grant access to users and require Azure MFA registration.
 -  You do not block & require MFA.

Azure AD Identity Protection

-  Available only in P2 in Azure AD.
- Uses adaptive machine learning algorithms and heuristics to detect anomalies and suspicious incidents that indicate potentially compromised identities.
 - Using this data, it generates reports and alerts that enable you to evaluate the detected issues and take appropriate mitigation or remediation actions
- Enables you:
 - Detect potential vulnerabilities affecting your organization's identities.
 - Configure automated responses to detected suspicious actions that are related to your organization's identities.
 - Investigate suspicious incidents and take appropriate action to resolve them.

Privileged Identity Management (PIM)

- Requires users to request for elevation to do a task.
 - Per elevated role / elevations, it allows to:
 - Set-up have notifications
 - Create incident/request tickets.
 -  Require MFA
 - Require approval
 - Set maximum activation duration (hours)
- Lists of users assigned
 - privileged roles to manage Azure resources
 - Administrative roles in Azure AD.
- Shows history of administrator activation
 - Including changes made to resources by admins.

- Alerts about changes in administrator assignments.
- Allows to require approval to activate Azure AD privileged admin roles.
- Enables on-demand, "just in time" administrative access to Microsoft Online Services like Office 365 and Intune, and to Azure resources such as management groups or Virtual Machines.
- Allows to review membership of administrative roles and require users to provide a justification for continued membership.
- 🚫 ! Available only in P2 in Azure AD.
- **Privileged accounts**
 - Accounts that administer and manage IT systems.

PIM Management

- Management of administrative roles with two branches:
 - Directory**
 - On Azure portal: Azure AD -> PIM -> Azure AD Roles
 - You can manage the users assigned to the built-in Azure AD organizational roles, such as *Global Administrator*.
 - Role assignments
 - Eligible:** They can activate the role when they need to perform privileged tasks.
 - Permanent:** Role is always activated.
 - Both assignments are permanent.
 - RBAC**
 - On Azure portal: Azure RBAC -> PIM -> Azure Resources
 - For consumption & spending
 - You can manage the users and groups assigned via Azure RBAC roles, including *Owner* or *Contributor*.
 - Role assignments
 - **Eligible**
 - Active**
 - Same as directory but permanent is called active because in PIM Eligible/Active can have timer where assignment is removed
 - Enable subscription in PIM: PIM -> Azure resources -> Discover resources
 - !** Once turned on, it cannot be turned off
 - The reason is that it in background PIM adds a principal on the scope it's on with User Access Administrator role
- Role assignments can be done on both Group and User level

- **!** If a Group is **Eligible** when assignment becomes **Active** then the privileges for whole Group is elevated instead of single individual, making harder to track individual requests.

Azure AD Domain Services

- Directory-aware applications may rely on LDAP or Windows Integrated Authentication (Kerberos or NTLM authentication)
- Line-of-business (LOB) applications running on Windows Server are typically deployed on domain joined machines, so they can be managed securely using Group Policy.
- To 'lift-and-shift' on-premises applications to the cloud, these dependencies on the corporate identity infrastructure need to be resolved.
- Administrators often turn to one of the following solutions to satisfy the identity needs:
 - Deploy a site-to-site VPN connection between workloads running in Azure IaaS and the corporate directory on-premises.
 - **!** Vulnerable to transient network glitches or outages
 - Extend the corporate AD domain/forest infrastructure by setting up replica domain controllers using Azure virtual machines.
 - Deploy a stand-alone domain in Azure using domain controllers deployed as Azure virtual machines.
- **!** All these approaches suffer from high cost and administrative overhead.
 - It requires administrators to deploy domain controllers using virtual machines in Azure.
 - Additionally, they need to manage, secure, patch, monitor, backup, and troubleshoot these virtual machines

OAuth2 Flow

1. Register your application with your AD tenant
 - App Registrations -> New application registration
 - Provide sign on URL for web apps, reply URI for API app
 - Get application client id to later use in code.
 - Applications -> Application -> Publish scope
 - Define the permissions (scopes) that can be granted to other applications
 - Add Scope Values as necessary (for example, "read")."
2. Authorize
 - User enters credential with consent to permissions
 - App goes to /oauth2/authorize on AD and get authorization token.
3. Access token

- App sends token to /oauth2/token
- Azure AD returns an access token upon a successful response
- 4. App validates access_token and return secure data to app
- 5. After a while token expires
 - App requests to /oauth2/token with refresh_token
 - AD gives new refresh_token and new access_token

2.1. SaaS services in Azure - Cognitive Services

Cognitive Services

- Set of APIs, SDKs and services available to developers to make their applications more intelligent, engaging and discoverable.
- Expands on Microsofts machine learning APIs and enables developers to easily add intelligent features – *such as emotion and video detection; facial, speech and vision recognition; and speech and language understanding* – into their applications.
- E.g.
 - **Agent:** Cortana
 - **Applications:** Office 365, Dynamics 365, SwiftKey, Pix, Customer Service and Support
 - **Services:** Bot Framework, Cognitive Services, Cortana Intelligence, Cognitive Toolkit
 - **Infrastructure:** Azure Machine Learning, Azure N Series, FPGA

Bing APIs

- **Bing Web Search**
 - Similar to Bing.com/search
 - The results include Web pages and may also include images, videos, and more.
- **Bing Image Search**
 - Similar to Bing.com/images
 - Returns images
- **Bing Autosuggest**
 - Lets you send a partial search query term to Bing and get back a list of suggested queries that other users have searched on.

LUIS

- **Intent detection:** Receive user input in natural language and extract meaning from it.
 - You can start with a prebuilt domain model, build your own domain specific model, or blend pieces of a prebuilt domain with your own custom information
- Once the intentions are identified (e.g. *Book Flight* or *Contact Help Desk*), you supply example phrases called utterances for the intents. Then you label the utterances with any specific details you want LUIS to pull out of the utterance.
- **Flow** : Create your own LU model => Train by providing examples => Deploy to an HTTP endpoint and activate on any device => Maintain model

Intents

- Purpose or goal expressed in a user's input.
- E.g. booking a flight, paying a bill, or finding a news article.
- You define and name intents that correspond to these actions.

Utterances

- An utterance is text input from the user that your app needs to understand.
- E.g. "Book a ticket to Paris", or a fragment of a sentence, like "Booking" or "Paris flight."
 - ! Utterances aren't always well-formed, and there can be many utterance variations for a particular intent.

Entities

- An entity represents detailed information that is relevant in the utterance.
- E.g. in the utterance "*Book a ticket to Paris.*", "Paris" is a location.
- By recognizing and labeling the entities that are mentioned in the user's utterance, LUIS helps you choose the specific action to take to answer a user's request.

Cognitive APIs

- **Text Analytics API:** Natural language processing over raw text.
 - Sentiment analysis
 - Key phrase extraction
 - Language detection
- **Speaker Recognition API:** algorithms for speaker verification and speaker identification.
- **Content Moderator API:** tracks, flags, assesses, and filters out offensive and unwanted content that creates risk for applications.
- **Face API**

- Face verification, finding similar faces, face grouping, and person identification.
- Image can be specified by file in bytes or valid URL.
- The API returns a face rectangle (left, top, width and height) indicating the face location in the image is returned along with each detected face.
- Optionally, face detection extracts a series of face related attributes such as pose, gender, age, head pose, facial hair and glasses.
- Face recognition is widely used in many scenarios including security, natural user interface, image content analysis and management, mobile apps, and robotics.

2.2. SaaS services in Azure - Bots (Bot Services, QnA Maker)

Bots

Bot services

- PaaS
- Provides an integrated environment that is purpose-built for bot development, enabling you to build, connect, test, deploy, and manage intelligent bots from one place.
- You can write a bot, connect, test, deploy, and manage it from your web browser with no separate editor or source control required.
 - For simple bots, you may not need to write code at all.
- The code glues in an HTTP REST endpoint the following:
 - Platform: Platform Services
 - AI: Intelligent Tools
 - SDK: Bot Framework SDK

Bot services key concepts

- **Multiple language support**
 - Leverages **Bot Framework SDK** with support for .NET and Node.js.
- **Bot templates**
 - E.g.:
 - Forms bot for collecting user input
 - a Language understanding bot that leverages LUIS to understand user intent
 - a QnA bot to handle FAQs
 - a Proactive bot that alerts users of events.

- **Bring your own dependencies:** Support NuGet and NPM.
- **Flexible development options**
 - Publish from Visual Studio
 - Code bot right in the Azure portal
 - Set up continuous integration and deploy the bot through GitHub, Visual Studio Team Services, and other supported development tools.
- **Connect to channels**
 - Bot Service supports popular channels for connecting your bots and the people that use them.
 - E.g. Skype, Facebook, Teams, Slack, SMS, and others.
- **Tools and services**
 - **Bot Framework Emulator:** Allows to test bots.
 - **Channel Inspector:** Allows to preview bots on different channels with the .
- **Open source**
 - The Bot Builder SDK is open-source and available on [GitHub](#).

QnA Maker

- Trains AI to respond to user's questions in a more natural, conversational way.
- Provides a GUI that allows non-developers to train, manage, and use the service for a wide range of solutions.
- Extracts a knowledge base from two types of input: FAQ pages (web pages or documents) and product manuals (PDF).
 - Once extracted, the QnA Maker service creates a **knowledge base** and bot using the knowledge base
- Handles indexing and ranking
- It can be consumed through REST API
- Over time, the knowledge base can be updated, retrained, and republished to meet the morphing needs to a user-facing web application.

2.3. SaaS services in Azure - Azure Machine Learning

Azure Machine learning

- Azure Machine learning is an end-to-end data science and analytics solution that's integrated into Azure.
- Built on top of open source technologies: Jupyter Notebook, Conda, Python, Docker, Apache Spark, and Kubernetes (also from Microsoft, e.g. Cognitive Toolkit)

- It allows users to develop experiments as well as deploy data and models via the cloud.
- Its composed of
 - Azure Machine Learning **Workbench**
 - Desktop application that includes command-line tools.
 - It allows users to help manage learning solutions via data ingestion and preparation, model development, experiment management,
 - Azure Machine Learning **Experimentation Service**
 - Helps handling the implementation of machine learning experiments
 - Provides project management, roaming, sharing, and git integration to support the Workbench.
 - Allows implementation of services across a range of environment options such as Local native, Local Docker container, or Scale out Spark cluster in Azure.
 - Creates Virtual environments for scripts to provide an isolated space with reproducible results.
 - Documents run history information
 - Visually displays the information so you can select the best model from your experiments.
 - Azure Machine Learning **Model Management Service**
 - Provides users the ability to deploy predictive models into a range of environments.
 - Information on models, such as the version and lineage, is notated from training runs throughout the deployment.
 - The models themselves are registered, managed, and stored in the cloud.
 - **MMLSpark (Microsoft Machine Learning Library for Apache Spark)**
 - Open-source Spark Package providing data science and Deep Learning tools for Apache Spark.
 - MMLSpark allows users to create robust, analytical, and highly scalable predictive models for large image and text datasets.
 - **Visual Studio Code Tools for AI**
 - Extension used with Visual Studio code that allows you to test, build, and deploy AI and Deep Learning solutions.
 - It contains various integration points from Azure Machine learning.
 - E.g. visualization of run history that displays the performance of training runs, select targets for your scripts to execute.
- Fully support various open source technologies, such as scikit-learn, TensorFlow, and more.
- Traditional BI flow: *(value & amount of information increases in each step)*
 - **Descriptive analytics:** What happened?
 - Leads to hindsight

- **Diagnostic analytics:** Why did it happen?
 - Leads to insight
- **Predictive analytics:** What will happen?
 - Leads to optimization & foresight
- **Prescriptive analytics:** How can we make it happen?

2.4. SaaS services in Azure - Media Processing

Media Processing

Media Services

- Extensible platform that enables developers to build scalable media management and delivery applications.
- It is based on REST APIs that enable you to securely upload, store, encode, and package video or audio content for both on-demand and live streaming delivery to various clients (for example, TV, PC, and mobile devices).
- Should be used with Content Delivery Network (CDN)
- Supports:
 - Secure Media, Encoding, On-Demand origin, Live ingest, Live Origin, Advertising, Media Job Scheduling, Static/Dynamic Packaging, Content Protection, Live Encoding, Analytics, Identity Management.
- Also partner technologies: Media processors, origin servers, live encoders etc.
- Packaging
 - **Static packaging** (traditional)
 - Have different assets (files) for different protocols.
 - Eg. HLS for HLS protocol (apple, mac)
 - Eg. Smooth for Smooth Protocol (XBOX, Windows)
 - Eg. MP4 for HTTPS
 - **Dynamic packaging**
 - MP4 asset can be automatically adopted to those protocols.

Computer Vision API

- API for advanced algorithms for processing images and returning information.
- Use cases/features:
 - ***Tag images based on content.***

- Based on more than 2000 recognizable objects, living beings, scenery, and actions
- **Generate descriptions of the content**
 - A collection of content tags forms the foundation for an image 'description' displayed as human readable language formatted in complete sentences.
 - Various descriptions are evaluated and a confidence score is generated.
- **Color schemes**
 - The colors are analyzed in three different contexts: foreground, background, and whole. They are grouped into twelve 12 dominant accent colors. Those accent colors are black, blue, brown, gray, green, orange, pink, purple, red, teal, white, and yellow. Depending on the colors in an image, simple black and white or accent colors may be returned in hexadecimal color codes.
- **Optical Character Recognition (OCR)**
 - Identify printed text found in images
 - You can use the result for search and numerous other purposes like medical records, security, and banking
- Other features include:
 - Categorize images
 - Identify the type and quality of images
 - Detect human faces and return their coordinates
 - Recognize domain-specific content
 - Flag adult content
 - Crop photos to be used as thumbnails
 - Recognize handwritten text
 - Distinguish color schemes

3.1.1. Storage - Azure Storage

Azure Storage

- Massively scalable to support big data scenarios & small amount of data.
- Access via REST API or SDKs: .NET, Java/Android, Node.js, PHP, Ruby, Python, PowerShell.
- Auto-partitioning with automatically load-balancing based on traffic.
- Billing is calculated by usage, not capacity.
- Elastic & decoupled from application.

- All resources in Storage can be protected from anonymous access and can be used in the Valet-Key pattern.
 - **Valet-Key Pattern**
 - Access resources using valet key (=key with granular & time-limited access for pre-defined operations/scopes).
 - E.g. SAS

Storage Types

Azure Blobs

- VHDs and large blocks of data (e.g. images or documents)
- 🗑️ **Soft delete** allows you to have a retention policy for files up to X days after deletion up to 365 days.
- Entities
 - **Page blobs**
 - Lend themselves to storing random access files **up to 8TB** in size
 - Ideal for VHD storage for Virtual Machines.
 - **Block blobs** are for images or other documents and files **up to 4TB** in size.
 - **Append blobs** are similar in format to block blobs but allow append operations.
 - Ideal for auditing and logging applications such as log or monitoring data from many sources.

VM disks

- Azure IaaS VMs can attach OS and Data Disks.
- Disks can be premium storage (SSD based) or Standard storage (HDD).
- ! Each storage account has a limit of 20000 IOPS throughput.
 - 💡 To be able to provide maximum performance with multiple disks => those data disks across many storage accounts.
- **Deployment**
 - Through Portal, PowerShell or Azure CLI.
 - You can deploy as top-level disk resource that'll be attached later or can create within VM creation template

Azure Disk Encryption

- Azure Disk Encryption is a capability that helps you encrypt your Windows and Linux IaaS VM disks.

- Disk Encryption leverages the industry standard BitLocker feature of Windows and the DM-Crypt feature of Linux to provide volume encryption for the OS and data disks.
- The solution is integrated with Azure Key Vault to help you control and manage the disk-encryption keys and secrets.
- The solution also ensures that all data on the VM disks are encrypted at rest in your Azure storage.

Disk types

Unmanaged disks

- Available in *Standard* and *Premium* tiers.
- Any Azure VM size can have multiple standard disks attached.
- A storage account can only store one type of disk.
- Can have disk snapshots
- ! Maximum of 20,000 IOPS per account, managed disk has not IOPS limit.

Managed disks

- Simplifies the creation and management of Azure IaaS VM Disks.
 - Azure manages the storage accounts.
 - Only choose storage type and the disk size.
- Other benefits:
 - Upgrade a standard disk to a premium disk and downgrade a premium disk to a standard disk.
 - ! requires you to detach the disk from a VM before it's upgraded or downgraded.
 - The ceiling of 20000 IOPS disappears without needing to manage additional storage accounts.
 - ! Allows up to 10000 VM disks per subscription.
 - In VM Scale Sets allows up to 1000 VMs per scale set using a Marketplace image.
 - Better reliability for Availability Sets
 - All the disks in a VM are stored in the same scale unit
 - Each VMs disks will be stored in separate scale units
 - 99.9% availability
 - Granular access control
 - **Managed disk snapshots**

- Stand-alone objects and can be used to create new disks.
- 💡 You are only billed for the used size of the data on the disk.
 - E.g. if a 4 TB disk that holds 500 GB of data, the snapshot will be billed for 500 GB.
- **Images:** Managed Disks support creating a managed custom image
- **Images vs. snapshots**
 - An image is a copy of the VM and all disks attached.
 - A snapshot is a single disk copy.
 - A snapshot is not a suitable choice for the scenario with multiple striped data disks. No snapshot co-ordination is available.
- **Managed Disks and Encryption**
 - By default Managed disks are encrypted by **Azure Storage Service Encryption**
 - Provides encryption at rest for disks, snapshots, and images
 - Also available at the VM level
 - In Windows it uses BitLocker Drive Encryption
 - **Azure Key Vault** integration is included which allows users to bring their own disk encryption keys.
- Sizing and pricing

Type	Allowed disks	Pricing
Premium	P4 (32 GB) -> P50 (4 TB)	Pay for allocated size without transaction charges
Standard	S4 (32 GB) -> S5 (4 TB)	Billed only for the number of transactions performed

Azure Tables

- Structured data (a NoSQL store)
 - Schema-less entities with strong consistency.
 - Entity group transactions for atomic batching.
 - Best for Key/value lookups on partition key and row key.
- 💡 Ideal for: User, device and service metadata, structured data.
- Supports OData protocol.
- Tables scale as demand increases.

- No limits on number of table rows or table size.
- Dynamic load balancing of table regions.
- Components
 - **URL format:** `http://<storage account>.table.core.windows.net/<table>`
 - **Storage Account** : All access to Azure Storage is done through a storage account.
 - **Table**
 - Collection of entities
 - Don't enforce a schema on entities
 - which means a single table can contain entities that have different sets of properties.
 - The number of tables that a storage account can contain is limited only by the storage account capacity limit.
 - **Entity** : An entity is a set of properties, similar to a database row. An entity can be up to 1 MB in size.
 - **Properties**
 - Name-value pair.
 - Each entity can include up to 252 properties to store data
 - Each entity also has 3 system properties:
 - Partition key
 - Row key
 - Timestamp


Partitioning

- Rate limit
 - **!** A partition has a scalability target of 500 entities per second.
 - The throughput may be higher during minimal load on the storage node, but it will be throttled down when the node becomes hot or very active.
- PartitionKey & RowKey properties
 - Can store up to 1 KB of string values.
 - Empty strings are also permitted; however, null values are not.
 - **Clustered index sorting:** Ascending PartitionKey then by ascending RowKey.
 - The sort order is observed in all query responses.
 - Each partition server can serve one or more partitions.
 - You should use more partitions, so that the Azure Table service can distribute the partitions to more partition servers.

Azure Queues

- For message passing in applications and for backlog processing
- Ideal for: Data sharing, Big Data, Backups. Functions
- Functions:
 - Decouple components and scale them independently.
 - Scheduling of asynchronous tasks.
 - Building processes/work flows.
 - No limits on number of queues or messages.
 - Message visibility timeout to protect from component issues.
 - **UpdateMessage** to checkpoint progress part way through.

Azure Files

-  Shared storage for applications using the standard SMB 3.0 protocol.
- Access data
 - **VMs and cloud services:** mounted shares via file I/O APIs.
 - **On-premises:** in a share via the File storage API.
 - E.g. on linux you run `sudo mount`
- Typical uses
 - Migrating on-premises applications that rely on file shares.
 - Transfer file between vm in different subnet without vpn.
 - Storing shared application settings, for example in configuration files.
 - Storing diagnostic data such as logs, metrics, and crash dumps in a shared location.
 - Storing tools and utilities needed for developing or administering Azure virtual machines or cloud services.

Azure Files Components

- **Storage Account**
- **Share**
 - SMB 3.0 file share in Azure.
 - All directories and files must be created in a parent share.
 - An account can contain an unlimited number of shares.
 - A share can store an unlimited number of files, up to the capacity limits of the storage account.
- **Directory:** An optional hierarchy of directories.

- **File:** A file in the share. A file may be up to 1 TB in size.
- **URL**
format: `https://[account].file.core.windows.net/<share>/<directory/directories>/`
 - E.g. `http://[account].file.core.windows.net/logs/CustomLogs/Log1.txt`

Azure File Sync

- On-premises -> File share in Azure
 - Turns your file server into a cache of the Azure-based file share.
 - Centralizes your file shares in Azure Files, whilst maintaining the compatibility of an on-premises file server.
 - Any protocol installed on the Windows Server can access the file share, including SMB, NFS, and FTPS.

Components

Storage Sync Service

- Azure resource created to host the Azure File Sync service.
- Required since service can sync between multiple storage accounts, hence an additional resource is required to manage this.
- A subscription can contain multiple storage sync services.

Sync Group

- Defines and controls the hierarchy and topology of the files to be synced.
- The sync group will contain *Cloud* and *Server* endpoints.
- Async service can contain multiple sync groups.

Registered Server

- Before adding a server endpoint to a sync group, the server must be registered with a storage sync service.
- A server can only be registered to a single sync service.
- Async service can host as many registered servers as you need.

Azure File Sync Agent

- To register a server, you need to install the Azure File Sync Agent.
- This is a small downloadable MSI package comprising three components:

- FileSyncSvc.exe: Monitors changes on Server Endpoints, and for initiating sync sessions to Azure.
- StorageSync.sys: A file system filter, which handles tiering files to Azure Files.
- PowerShell Management cmdlets: PowerShell cmdlets for the Microsoft.StorageSync Azure resource provider.

Server Endpoint

- Once registered, you can add a server to a Sync group, this then becomes a server endpoint.
- A server endpoint is synonymous with a folder or a volume on the server that will cache the contents of the Azure File Share.
- Cloud tiering is configured individually by server endpoint.

Cloud Endpoint

- When added to a sync group, an Azure File Share is a cloud endpoint.
- One Azure File Share can only be a member of one Cloud Endpoint and thereby can only be a member of one Sync Group.
- Any files that exist in a cloud endpoint or a server endpoint before they are added to the sync group, automatically become merged with all other files in the sync group.

Features

- **Adding files to the File Share**
 - Adding and removing files directly within the Azure file share.
 - Syncs every 24 hours as detection job runs every 24 hours.
 - Only from on-prem to azure
- **Cloud tiering**
 - Caching mechanism to save space.
 - Flow:
 - a. File is tiered
 - b. The sync system filter replaces the local file with a pointer to the location if the Azure file share.
 - c. When accessed locally the file is downloaded and opened for use, e.g. Hierarchical Storage Management (HSM).
- **Supported versions of Windows Server:** Windows Server 2012 R2 and Windows Server 2016
- **Access control lists (ACL)**
 - Supported and enforced on files held on Server endpoints.
 - **!** Azure Files do not currently support ACLs.

- **NTFS compression**
 - Fully supported, and Sparse files are fully supported but are stored in the cloud as full files, and any cloud changes are synced as full files on server endpoints.
- **Failover Clustering**
 - Supported for File Server for *General Use* but not for *Scale-out file server* for application data.
 - **!** *Cluster Shared Volumes* are not supported.
 - To function correctly, the sync agent must be installed on every node of a cluster.
- **Data Deduplication**
 - Fully supported for volumes that do not have cloud tiering enabled.
- **Encryption solutions**
 - Azure File Sync, is known to work with BitLocker Drive Encryption and Azure Rights Management Services.
 - NTFS Encrypted File System does not work with Azure File Sync.

Storage Performance and Pricing

1. **Standard**
2. **Premium**
 - SSD disk support for VMs.
 - Using multiple disks gives your applications up to 256 TB of VM storage
 - Up to 80,000 I/O operations per second (IOPS) per VM, and a disk throughput of up to 2,000 megabytes per second (MB/s) per VM.
 - Only LRS redundancy
 - Applications that require consistent high performance and low latency such as SQL Server, Oracle, MongoDB, MySQL, and Redis can run happily in Azure Premium Storage.
 - More expensive: The storage is allocated on creation and is charged for the whole disk size rather than the data used.

Disk Types

Disk Type	Description	💡 Suggested Use
Azure Files	SMB 3.0 interface, client libraries and a REST interface access from anywhere to stored files.	Application lift and shift using native file system API, Windows File Shares.

Disk Type	Description	💡 Suggested Use
Azure Blobs	Client libraries, REST interface, for unstructured data stored in Block blobs	Streaming and random access, access application data from anywhere.
Azure Disks	Client libraries, REST interface for persistent data accessed from a VHD,	Access to data inside a Virtual machine on a VHD, lift and shift file system API apps that write to persistent disks.

3.1.1.1. Storage - Storage Account

Storage Account

- Stores & manages storage services.
 - 💡 Managed Disks for Virtual Machine disk storage does not need a storage account.
- Types

Type of Account	Services Supported	Types of Blobs supported
General Purpose Standard	Blob, File, Queue services	Block blobs, Page blobs and Append blobs
General Purpose Premium	Blob service	Page blobs
Blob Storage (hot and cool access tiers)	Blob service	Block blobs and Append blobs

Storage account security

- Access is controlled by **Storage Account Keys**
 - There are primary and secondary to allow for key refreshes whilst maintaining access.

- If you have the key, you have access to the account and all the data in the account.

Container Security

- Provided for blob storage at the time of creation; the container can be **public**, **private** or **read-only**.
 - Public read access policy
 - Controls what data is available anonymously for your container
 - Setting can be *Container*, *Blob* or *Off*.

Setting	Individual blobs and their properties can be accessed	Blobs can be enumerated
Container	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>
Blob	✓ <input type="checkbox"/>	✗
Off	✗	✗

Azure AD RBAC

- 💡 Use for:
 - Management functions on Storage Account
 - Storage account keys

Shared Access Signatures

- A URI that grants restricted access rights to containers, blobs, queues, and tables.
- For delegations, not resources.
- Access is for specified period of time, with a specified set of permissions.
- All information is described in URI.
 - E.g.
GET https://[account].blob.core.windows.net/pictures/profile.jpg?sv=2012-02-12&st=2009-02-09&se=2009-02-10&sr=c&sp=r&si=YWJjZGVmZWw%3d%3d&sig=dD80ihBh5jfNpym05Hg1IdiJIEvHcJpCMiC MnN%2fRnbI%3d

- The signature is using version 2012-02-12 of the storage API; it allows read access is beginning from 02/09/09 to 02/10/09 to the container.
- You can grant following:
 - **Content:** Reading and writing page or block blob content, block lists, properties, and metadata
 - **Deleting:** Deleting, leasing, and creating a snapshot of a blob
 - **Listing:** Listing the blobs within a container
 - **CRUD**
 - Adding, removing, updating, and deleting queue messages
 - Querying, adding, updating, deleting, and upserting table entities
 - **Metadata:** Getting queue metadata, including the message count
 - **Copying:** Copying to a blob from another blob within the same account

Shared Access Policies

- A shared access policy adds the ability to revoke, expire or extend access.
- Settings defined in a server-stored policy can be changed and are reflected in the token without requiring a new token to be issued, but settings defined in the token itself cannot be changed without issuing a new token.
- This approach also makes it possible to revoke a valid SAS token before it has expired.
 - To revoke a stored access policy, you can either delete it or rename it by changing the signed identifier.
- **!** A maximum of five access policies may be set on a container, table, or queue at any given time.

Storage account replication

- **Locally redundant storage (LRS)**
 - Replicated three times across two to three facilities, either within a single region or across two regions
 - Do not protect from the failure of a single facility.
- **Zone-redundant storage (ZRS)**
 - Replicated three times across two to three facilities, either within a single region or across two regions.
- **Geo-redundant storage (GRS)**
 - Replicated three times within the primary region, and is also replicated three times in a secondary region hundreds of miles away from the primary region.
 - Replicates asynchronously
 - Means replica is eventually consistent and could possibly have older data if you access the replica before the replication

- **Read access geo-redundant storage (RA-GRS)**

- Provides read-only access to primary location on top of GRS.

Replication	LRS	ZRS	GRS	RA-GRS
Data stored in multiple datacenters	No	Yes	Yes	Yes
Data read from secondary & primary location	No	No	No	Yes
No of copies of data stored in separate nodes	3	3	6	6

3.1.2. Storage - StorSimple

StorSimple

- A physical device.
- Creates workflows for migrating data to a cloud storage center or back on premise.
- Combination of service device management tools.
- On-premises hybrid storage array
- Manages communication with cloud storage
 - Helps to ensure the security and confidentiality of data
- Includes
 - Solid state drives (SSDs)
 - Hard disk drives (HDDs)
 - Support for clustering and automatic failover.
 - Shared processor, shared storage, and two mirrored controllers
- You can alternatively use StorSimple to create a virtual device that replicates the architecture and capabilities of the actual hybrid storage device.
 - The StorSimple virtual device (also known as the **StorSimple Virtual Appliance**) runs on a single node in an Azure virtual machine.
- StorSimple provides a web-based user interface (the StorSimple Manager service), or you can use PowerShell CLI.
- Security through encryption algorithms to protect data stored in or traveling between the components of StorSimple solution.

Transparent integration

- 🗄️ Uses Internet Small Computer System Interface (iSCSI) protocol to invisibly link data storage facilities.
 - iSCSI (Small Computer System Interface)
 - Storage networking standard for linking data storage facilities over TCP/IP
- Data that's stored in the cloud, in the data center, or on remote servers, appears to be stored at a single location.

Reduced storage costs

- Compression
- **Deduplication**
 - 🗄️ Eliminates redundant versions of the same data (*deduplication*)

Simplified storage management

- Provides system administration tools that you can use to configure and manage data:
 - Backup and restore functions from a *Microsoft Management Console (MMC)* snap-in.
 - Separate, optional interface to extend StorSimple management and data protection services to content stored on SharePoint servers.

Improved disaster recovery and compliance

- Does not require extended recovery time. Instead, it restores data as it is needed.
 - Regular operations can continue with minimal disruption.
- You can configure policies to specify backup schedules and data retention.

Data mobility

- Data uploaded to Microsoft Azure cloud services can be accessed from other sites for e.g. recovery and migration purposes.
- You can use StorSimple to configure StorSimple virtual devices on virtual machines (VMs) running in Microsoft Azure.
 - The VMs can then use virtual devices to access stored data for test or recovery purposes.

Data Tiering

- Automatically tiers and classifies your data.

- Based on how often you access it.
- Data is always being shuffled between tiers as the mechanism learns about your usage patterns.
 - To enable quick access, it stores hot data
 - On SSD.
 - Locally
 - It stores occasionally used (warm data) data
 - on HDDs in the device or on servers at the data center.
 - Inactive data
 - Automatically migrates to the cloud.
- Rearranges data and storage assignments as usage patterns change

3.2. SQL (Azure SQL Database)

SQL (Azure SQL Database)

- Other third-party managed SQL databases in Azure include:
 - Azure Database for MySQL**
 - MYSQL community with tools preinstalled like mysql.exe and phpMyAdmin.
 - Azure Database for PostgreSQL**
 - Both has common features:
 - You can run one or more databases with this instance.
 - High availability with no additional cost.
 - Predictable performance, using inclusive pay-as-you-go pricing.
 - Scale on the fly within seconds.
 - Secured to protect sensitive data at-rest and in-motion.
 - Automatic backups and point-in-time-restore for up to 35 days.
 - Enterprise-grade security and compliance.
- Other options
 - Own VMs running e.g. MySQL
 - ClearDB provides managed MySQL that you can create from Azure Marketplace.

Azure SQL Database

- Database as a service
- Predictable performance
 - Through comparison of DTUs (Database Throughput Units)

- DTUs describe capacity of tier and performance level.
 - Relative: E.g. Basic (B) 5, Standard (S2) 50, so standard is 10 times power of Basic.
- High compatibility
 - A tabular data Stream (TDS) endpoint is provided for each logical server.
- Management
 - In portal: Azure Management Portal -> Manage
 - REST API, PowerShell or Xplat CLI
- **Retention**
 - **Long-term retention (LTR):** Automatically retain backups in Azure Blob storage for up to 10 years
 - **Azure SQL Database automatic backups:** 7-35 days

Database Tiers

- Three tier where every tier have different performance levels:

Tier	DTU	Ideal for
Basic	5	small dbs, single active operation, dev/test, small scale apps
Standard	10 - 100	multiple operations, workgroup or web apps
Premium	100 - 800	high transaction volumes, large number of users, multiple operations, mission critical apps

- Basic/Standard model
 - Based on remote storage.
 - Uses Azure Premium Storage Disks, i.e. accessed over network.
- Premium/Business Critical model
 - Uses AlwaysOn Availability Groups
 - Has local attached SSD storage
 - Provides higher IOPS and throughput

Elastic Scale

- Scaling it/out by simplified sharding.
 - Coordinates data movement between shards to split or merge ranges of data among different databases
- Satisfies common scenarios such as pulling a busy tenant into its own sharding
- Split-Merge service
 - Provided through a downloadable package
 - Customers can deploy as an Azure cloud service into their own subscription.
 - Two main parts:
 - An **Elastic Scale library (SDK)** for client applications to configure shards and access shards.
 - Direct transactions to the appropriate shard
 - Perform queries across multiple shards
 - Modify service tier for existing shards
 - The Elastic Scale features in Azure SQL Database that implements the any changes requested by your application.

Availability

- **Azure Site Recovery:** Only for VMs

AlwaysOn

- Available only in SQL servers
- Azure SQL database implements it in its underlying infrastructure for Premium tier to achieve high availability but abstracts it away using active geo-replication
- **AlwaysOn availability groups**
 - An availability group supports a failover environment for a discrete set of user databases, known as availability databases that fail over together.
- **Always On Failover Cluster Instances**
 - Leverages *Windows Server Failover Clustering (WSFC)* functionality
 - Provides local high availability through redundancy at the server-instance level failover cluster instance (FCI).
 - **FCI (failover cluster instance)**
 - Single instance of SQL Server that is installed across Windows Server Failover Clustering (WSFC) nodes and, possibly, across multiple subnets

Active geo-replication

- Leverages the Always On technology of SQL Server.
- Azure SQL Database feature.
 - Supported in Elastic Pools.
 - ! Not supported in managed instances.
 - ! Use auto-failover groups for them.
 - It also supports SQL database.
- Multiple secondaries are supported as opposed to auto-failover groups.
- Allows you to create readable secondary databases of individual databases on a SQL Database server in the same or different data center (region).
- Best practice configuration:
 - User => Azure Traffic Manager =>
 - a. Ingress LB => SQL in Primary Logical Server
 - b. Ingress LB => SQL in Secondary Logical Server
 - Capabilities:
 - Automatic Asynchronous Replication
 - Readable secondary databases
 - Planned / Unplanned failover
 - Multiple readable secondaries
 - Geo-replication of databases in an elastic pool
 - Each secondary database can separately participate in an elastic pool or not be in any elastic pool at all.
 - Configurable compute size of the secondary database
 - User-controlled failover and failback
 - Keeping credentials and firewall rules in sync

Auto-failover group

- Allows you manage replication and failover of a group of
 - databases on a SQL Database server
 - or all databases in a Managed Instance to another region
- It uses the same underlying technology as active geo-replication.
- You can failover
 - i. Manually
 - ii. Alternatively you can delegate it to the SQL Database service based on a user-defined policy.

- When working with single or pooled databases on a SQL Database server and you want multiple secondaries in the same or different regions, use active geo-replication.
- You get URL (HA URL) for the DB.
- Terminology and capabilities
 - **Failover group**
 - Group of databases that can either be:
 - SQL Database servers
 - Managed Instances
 - **Primary & secondary hosts**
 - Use Active-geo replication for multiple secondary hosts.
 - **Adding databases in elastic pool to failover group**
 - *Multiple failover groups*: You can configure multiple failover groups for the same pair of servers to control the scale of failovers. Each group fails over independently
 - ! Managed Instance does not support multiple failover groups
 - **Grace period with data loss**: By configuring `GracePeriodWithDataLossHours`, you can control how long the system waits before initiating the failover that is likely to result data loss.

Failover strategies

- **Automatic failover policy**: Configured by default
- **Read-only failover policy**:
 - Kicks in after a set period (hours) so that data is not last during a long failure.
 - Disabled by default
 - When disabled:
 - Performance of the primary is not impacted when the secondary is offline.
 - Read-only sessions will not be able to connect until the secondary is recovered
 - When enabled
 - Read-only traffic will be automatically redirected to the primary if the secondary is not available.
 - Performance gets lower in the primary
 - Use if you
 - Cannot tolerate downtime for the read-only sessions.
 - are OK to temporarily use the primary for both read-only and read-write traffic at the expense of the potential performance degradation of the primary
- **Planned failover**: Full synchronization without data loss
 - Use-cases:

- Perform disaster recovery (DR) drills in production when the data loss is not acceptable
- Relocate the databases to a different region
- Return the databases to the primary region after the outage has been mitigated (failback).
- **Unplanned failover:** Switches directly without any synchronization.
- **Manual failover:** You can initiate forced or friendly failover (with full data synchronization).

3.3. Big data services

Big data services

Azure Search

- PaaS / Search as a Service over scoped text content.
- Stores your data in an index that can be searched through full text queries.
 - Can be created in Azure Portal or with SDK/REST APIs.
 - Can be auto-generated from SQL Database or Cosmos DB.
- Search is not only indexing!
 - Search needs to be smarter, have language understanding, and understand uniqueness.
- Advanced search behaviors
 - Type-ahead query suggestions based on partial term input
 - Hit-highlighting
 - Faceted navigation
 - Natural language support using linguistic rules for the specified language.
- **Pricing**
 - Free: Shared with other Azure Search subscribers
 - Higher
 - Dedicated resources only used by your service
 - More search units & partitions
 - **Search unit**
 - How fast it indexes
 - Each partition and replica counts as one SU
 - E.g. 3 replicas + 3 partitions = 9 SU (as each partition have its own replica)

Azure Cosmos DB

- Multi-model database service.
- Globally distributed: users access data centers closer to them.
- Four different APIs:
 - Document DB (SQL) API,
 - MongoDB API
 - Graph (Gremlin) API
 - Tables (Key/Value) API
- **The Azure Cosmos DB Data Migration tool**
 - Open-source solution that imports data to Azure Cosmos DB
 - Sources, include: JSON files, MongoDB, SQL Server, CSV files, Azure Cosmos DB collections.

Consistency levels

- Automatic data distribution across partitions.
- Same partition key ensures data will be stored within same partition.
- 🐾 **Consistent-prefix**
 - Data versions can be behind by are always ordered in a right way based on when they're written.
- Four consistency levels from stronger guarantees to better performance and availability:
- From strongest to loosest: Strong => Bounded Stateless => Session => Eventual
- Consistency strategy
 - Stronger consistency -> Write is slower
 - Looser -> Performance is at peak but read can lag behind to older versions of data

Strong

- Write operation on primary database => it's replicated to the replica instances.
- The operation is only committed (and visible) on the primary after it has been committed and confirmed by ALL replicas.
- Always read most recent copy from master r/w node.

Bounded Stateless

- Reads honor consistent-prefix guarantee.
- Established through 2 metrics: time/version.

- E.g. goal: 90%>
- Difference from Strong is you can configure how stale documents can be within replicas.
 - **Staleness** : Quantity of time (or version count) a replica document can be behind the primary document.

Session

- Reads honor consistent-prefix guarantee.
- Guarantees all queries are served by same server per session.
 - You always see the latest change you make.
- All read & write are consistent & monotonic across primary and replica instances within a user session.

Eventual

- Commits any write operation against the primary immediately.
- Transactions are handled asynchronously and will eventually (over time) be consistent with the primary.
- Most performant as you don't wait for replicas to commit to finalize its transactions.

Azure Synapse Analytics

- Cloud-based Enterprise Data Warehouse (EDW)
- Formerly known as **SQL Data Warehouse**.
- Main difference from SQL is that it's underlying infrastructure is that it's a **columnar storage**.
 - Tables are still relational
- Quickly run complex queries across petabytes of data.
 - Leverages Massively Parallel Processing (MPP).
- E.g. use as a key component of a big data solution.
- Supports **PolyBase T-SQL queries**
 - PolyBase is a technology that accesses and combines both non-relational and relational data, all from within SQL Server.
 - It allows you to run queries on external data in Hadoop/Spark or Azure blob storage.
- 💡 Choose if you run many aggregated queries.
- Data flow
 - **Ingest**: Data orchestration and monitoring
 - **Store**: Big data store

- **Prep & train:** Hadoop/Spark and Machine Learning
- **Model & serve:** Data warehouse

Azure Data Lake Store

- Hyper-scale repository for big data analytic workloads.
- Stores data of any size, type and ingestion speed in a one single place for operational and exploratory analytics.
- It's a distributed file system
- Can be accessed from Hadoop (available with HDInsight cluster) using the WebHDFS-compatible REST APIs.
 - **WebHDFS** => *Web Hadoop File System* for IaaS.
 - Used also by Kafka.
 - **HDInsight**
 - Compute layer of the data lake store.
 - Managed hadoop cluster
- Includes security (e.g. OAuth, TTT), manageability, scalability, reliability and availability.

File systems

- **Single machine:** Fat NFS
- **Network file-systems:** NFS, SMB
 - Distributed lock.
 - Different nodes can mount same drive without corruption.
- **Cluster file-systems:** ZFS, ADFS, Oracle RAC
 - They respect each others lock.
 - Allows multiple machines to manipulate files within them concurrently.
 - Oracle RAC is **active-active**.
 - Machines share their caches.
 - They cross replicates to create illusion that they're active nodes, but the reality is that there's a single master node.
 - If master node fails -> mastership is delegated to another node.
- **Distributed file system**
 - Connects machines.
 - Instead of everyone mounting a shared device -> Every machine has a local storage and distributed across others.
 - Allows distributing jobs so it can be distributed.
 - Parallel processing, e.g. Hadoop.
 - It's not much about distributing files but parallel processing.

- **More**, e.g. storage based on blockchain.

Azure Data Lake Analytics

- Biggest challenge of Hadoop is that it's a very big ecosystem.
 - So many components, hard to get stuff done.
- Simplify big data analytics.
- Can handle jobs of any scale instantly by setting the dial for how much power you need.
- Pay only for your job when it is running, making it cost-effective.
- The analytics service supports Azure Active Directory letting you manage access and roles, integrated with your on-premises identity system.
- Includes **U-SQL**
 - Similar to C# and SQL.
 - A language that unifies the benefits of SQL with the expressive power of user code.
 - Scalable distributed runtime enables you to efficiently analyze data in the store and across SQL Servers in Azure, Azure SQL Database, and Azure SQL Data Warehouse.

Azure Data Factory

- Used for data integration
- Pipeline solution to schedule data-driven workflows.
 - Typically a pipeline is:
 - *Connect & Collect / Ingest*
 - *Store*: E.g. Azure Store
 - *Transform & Enrich*
 - Process in Spark, Data Lake Analytics and Machine Learning.
 - *Prep & train* with **Azure Databricks**.
 - *Publish*
 - Load into analytics engine (e.g. Azure SQL database) to be used by BI tools.
 - *Monitor*
 - Check failure/success rates
- Handles
 - Extract-transfer-load
 - Data integration
 - Can ingest data from data stores.
 - Send output data to data stores.

- E.g. file shares, FTP web, databases, SaaS services.
- Hybrid extract-transfer-load

3.4. Azure Automation

Azure Automation

- SaaS
- Provides a way for users to automate the manual, long-running, error-prone, and frequently repeated tasks that are commonly performed in a cloud and enterprise environment
- Automate processes using runbooks or automate configuration management using Desired State Configuration, in Azure, other cloud services, or on-premises.
- Schedule processes to be automatically performed at regular intervals
- Trigger through Azure Portal, Webhooks, PowerShell or Alerts.

Automation sandboxes

- Runbooks that you run in Azure are executed on Automation sandboxes.
- They are hosted in Azure PaaS virtual machines.
- They provide tenant isolation for all aspects of runbook execution – modules, storage, memory, network communication, job streams, etc.
- This role is managed by the service and is not accessible from your Azure or Azure Automation account for you to control.

Hybrid Runbook Worker (HRW) roles

- To automate the deployment and management of resources in your local datacenter or other cloud services, after creating an Automation account, you can designate one or more machines to run the Hybrid Runbook Worker (HRW) role.
- Each HRW requires the Microsoft Management Agent with a connection to a Log Analytics workspace and an Automation account.
- Log Analytics is used to bootstrap the installation, maintain the Microsoft Management Agent, and monitor the functionality of the HRW
- The delivery of runbooks and the instruction to run them are performed by Azure Automation.

Automation Account

- You first create an Automation Account object.
- It includes following objects to be used with runbooks:
 - **Certificates**
 - Contains a certificate used for authentication from a runbook or DSC configuration or add them.
 - **Connections**
 - Contains authentication and configuration information required to connect to an external service or application from a runbook or DSC configuration.
 - **Credentials**
 - It is a PSCredential object
 - It contains security credentials such as a username and password required to authenticate from a runbook or DSC configuration.
 - **Integration modules**
 - They are PowerShell modules included
 - Make use of cmdlets within runbooks and DSC configurations.
 - **Schedules**
 - Schedules start or stop a runbook at a specified time, including recurring frequencies.
 - **Variables**
 - Contain values that are available from a runbook or DSC configuration.
 - **DSC Configurations**
 - PowerShell scripts that describes how to configure an operating system feature or setting or install an application on a Windows or Linux computer.
 - **Runbooks**
 - They are a set of tasks that perform some automated process in Azure Automation based on Windows PowerShell.

Authentication

- Role-based access control is available with Azure Resource Manager to grant permitted actions to an Azure AD user account and Run As account, and authenticate that service principal.
- When you create authentication account, it comes with two authentication entities:
 - **A Run As account**
 - Creates a service principal in Azure Active Directory (Azure AD) and a certificate.

- Assigns the **Contributor** role-based access control (RBAC), which manages Resource Manager resources by using runbooks.
- **A Classic Run As account.**
 - This account uploads a management certificate, which is used to manage classic resources by using runbooks.
- You can add your own connection with a certificate or as service principal.
- For more access to VM you can add extensions to the automation account to e.g. ssh into machine and run a script.

Cross-cloud and regions

- Azure Automation lives in Azure
 - can run tasks on different platforms
 - but are triggered from Azure Automation.
- Configured across regions.
- Allows management and configuration of
 - Azure systems
 - on-premises using **Azure Automation Hybrid Worker**
 - or systems in AWS, Google Cloud, or any other 3rd party hosting data center using **Azure Automation Agent**.

Configuration Management

- Typical flow: You **provision/manage infrastructure** through **bootstrap agents** that **customizes VMs**.
- You can also use infrastructure automation tools such as Chef and Puppet in Azure:

Chef

- Automation platform that helps define how your infrastructure is configured, deployed, and managed.
- Additional components includes:
 - **Chef Habitat** for application lifecycle automation
 - **Chef InSpec** helps automate compliance with security and policy requirements.
- *Chef Clients* are installed on target machines, with one or more central *Chef Servers* that store and manage the configurations.

Puppet

- Handles the application delivery and deployment process.
- Agents are installed on target machines to allow *Puppet Master* to run manifests that define the desired configuration of the Azure infrastructure and VMs.
- Puppet can integrate with other solutions such as Jenkins and GitHub for an improved devops workflow.

3.5. Data Analysis (Azure Analysis Services, HDInsight, Azure Data Catalog)

Data Analysis

Azure Analysis Services

- PaaS
- Integrated with Azure data platform services.
- You can mashup and combine data from multiple sources, define metrics, and secure your data in a single, trusted semantic data model.
- Handles
 - Security
 - In-memory cache
 - Data modeling
 - Lifecycle management
 - Business logic & metrics
- Compatible with many features already in *SQL Server Analysis Services Enterprise Edition*
 - Supports tabular models at the 1200 and 1400 compatibility levels
 - Partitions, row-level security, bi-directional relationships, and translations are all supported.
 - In-memory and DirectQuery modes are also available for fast queries over massive and complex datasets.

Integrations

- Data Sources
 - **Cloud:** E.g. SQL Database, Azure Synapse Analytics, Data Lake, HDInsights/Spark...
 - *On-premises:* E.g. SQL Server / Oracle...

- Client tools
 - **Cloud:** Power BI
 - **On-premises:** Third-Party. Power BI Desktop. Excel

Tabular Object Model (TOM)

- Client library for SQL to describe model objects for developers.
- Exposed in JSON through the Tabular Model Scripting Language (TMSL) and the AMO data definition language.
 - TOM is built on AMO.
 - **Analysis Management Objects (AMO)** is a library of programmatically accessed objects that enables an application to manage an Analysis Services instance.
 - E.g. AMO has data mining classes
- Has classes for models, relationship, roles, annotations, cultures etc. to manage SQL analysis objects.
- Structured in a tabular form.
- Arranges data elements in vertical columns and horizontal rows. Each cell is formed by the intersection of a column and row.

HDInsight

- Common use:
 - Create HDInsight
 - Schedule Jobs
 - Delete HDInsight Cluster
- Azure distribution of Apache Hadoop components
 - Framework for processing and analysis of big data sets on clusters.
 - Including Apache Hive, HBase, Spark, Kafka, Storm, R and many others.
 - Apache Spark is an open-source parallel processing framework that supports in-memory processing to boost the performance of big-data analytic applications.
- Built on top of Azure Storage

Azure Data Catalog

- A single, central place for all of an organization's users to contribute their knowledge and build a community and culture of data.
 - It includes a crowdsourcing model of metadata and annotations.

- Descriptive metadata supplements the structural metadata (such as column names and data types) that's registered from the data source.
 - The data remains in its existing location, but a copy of its metadata is added to Data Catalog, along with a reference to the data-source location.
 - The metadata is also indexed to make each data source easily discoverable via search and understandable to the users who discover it.
- Any user (analyst, data scientist, or developer) can discover, understand, and consume data sources.
 - Users can contribute to the catalog by tagging, documenting, and annotating data sources that have already been registered.
 - They can also register new data sources, which can then be discovered, understood, and consumed by the community of catalog users.

3.6. Backup (Azure Backup & Azure Site Recovery)

Backup

Azure Backup

- Backup as a service (BaaS) solution with tools in the cloud and for on-premises
- Data is stored in a Storage Account.
 - **Azure Backup Server**
 - Can be installed on Azure or on-premises and can back up VMs.
 - Inherits much of the workload backup functionality from Data Protection Manager (DPM).
 - Backs up in Site Recovery vault.
 - Can back up File servers, SQL Server, Hyper-V, Exchange Server
- In a seamless portal experience with Azure Site Recovery, gives you cost-efficiency and minimal maintenance, consistent tools for offsite backups and operational recovery, and unified application availability and data protection.
- There are three primary options for backing up to Azure Backup:
 - i. Azure Backup / Restore of On-Premises Files & Folders
 - 💡 Ideal for backing up Files & Folders only
 - Steps:
 - a. Deploy the Azure Backup Agent (Azure Recovery Services Agent) on the VM guests running on-premises Hyper-V / SCVMM / VMware / Physical infrastructure.
 - b. Configure Azure Backup from within the VM guest.

- c. Configure the integration with Azure Backup Vault.
 - d. Run the Backup job from within the VM guest.
 - Runs on 443 HTTPS over Public Internet or ExpressRoute with Public Peering
 - e. Files & Folders backup will be stored in Azure Backup Vault, and can be restored from there.
- ii. Azure Backup / Restore of On-premises running full workloads (OS, Sysvol and Applications)
 - 💡 Better for backing up full system workloads (OS, system state, applications – consistent)
 - Steps:
 - a. Deploy the Azure Backup Server (or System Center DPM 2012 R2 or 2016) on the on-premises Hyper-V / SCVMM / Vmware / Physical infrastructure.
 - b. Configure Azure Backup Server backup policies, backup storage (2-tier) and deploy agents to your workloads.
 - c. Configure the integration with Azure Backup Vault.
 - d. Run the Backup job from within the Azure Backup Server console.
 - Runs on 443 HTTPS over Public Internet or ExpressRoute with Public Peering
 - e. VM workloads (system state, OS, applications,...) backup will be stored in Azure Backup Vault, and can be restored from there.
- iii. Azure VM Backup / Restore to Azure Backup Vault
 - 💡 Best for when you want to backup Azure VMs to Azure Backup Vault
 - Steps:
 - Deploy the Azure Backup Extension, or select Azure Backup in the VM configuration.
 - Configure Azure Backup backup policies, in the Azure platform.
 - Configure the integration with Azure Backup Vault.
 - Run the Backup job from within the Azure Platform.
 - Runs on 443 HTTP over Azure Backbone Infrastructure.
 - Azure VMs will be backed up as full VM snapshots, and can be restored from within the Azure Portal.
- Allows **application consistent backups** in Linux environments
 - ! You need to run a pre- and post- backup script.
 - The VM Snapshot will be your VM Backup
 - Stored in the Backup Vault using an incremental update process

Hybrid Backup Encryption

- Allows for end-to-end encryption of the backup platform:
- It starts with a passphrase for the Azure Recovery Services Agent installation.
- The next layer is the Backup Data itself, which gets encrypted in transit.
- Once the data is stored in Azure Backup Vault, it gets encrypted at rest as well.

Monitoring

- Azure Backup monitoring is possible from Log Analytics.
- Out of Log Analytics, one can get a detailed view on the backup statistics, the amount of data that is being consumed, successful and failed jobs and alike.
- Azure Backup allows for reporting integration with Microsoft Power BI.

Microsoft System Center Data Protection Manager (DPM)

- Underlying infrastructure of Azure Back-up
- Backs up both on-prem servers/VMs & cloud VMs
- Can store back-up data to:
 - Disk: For short-term storage DPM backs up data to disk pools.
 - Azure: DPM data stored in disk pools can be backed up to the Microsoft Azure cloud using the Azure Backup service
 - Tape
- Features:
 - **Application-aware backup:** Application-aware back up of Microsoft workloads, including SQL Server, Exchange, and SharePoint.
 - **File backup:** Back up files, folders and volumes for computers running Windows server and Windows client operating systems.
 - **System backup:** Back up system state or run full, bare-metal backups of physical computers running Windows server or Windows client operating systems.
 - **Hyper-V backup:** Back up Hyper-V virtual machines (VM) running Windows or Linux. You can back up an entire VM, or run application-aware backups of Microsoft workloads on Hyper-V VMs running Windows.
- **Microsoft Azure Backup Server (MABS)**
 - Can be installed on Azure or on-premises and can back up VMs.
 - Inherits much of the workload backup functionality from *Data Protection Manager (DPM)*.
 - Backs up in Site Recovery vault.
 - Can back up File servers, SQL Server, Hyper-V, Exchange Server

Azure Site Recovery

- Built to provide a data center disaster recovery solution for your VM workloads to Azure.
 - Recovers from
 - Hyper-V
 - VMware
 - Physical hosts
 - AWS VMs
- Replication-based failover to Azure Virtual Machines
- Failover & Failback
- Application-consistent failover
- Good as VM lift & shift migration and for dev/test scenarios
 - Full machine-state replication to an Azure VM
 - Zero-data loss during migration

Replication policies

- 🛠️ **Recovery Point Objective (RPO)** is the maximum time of acceptance for data loss.
- 🛠️ **Recovery Time Objective (RTO)** in ASR means the period when a failover starts to the time the process completes and a virtual machine is running in Azure.
- 🛠️ **App-consistent snapshots** capture disk data, all data in memory, and all transactions in process.

Running a failover

- On portal: Recovery Plans > recovery plan name > Click "Failover"
- You can alternatively run failover from "Replicated Items"
- 🛠️ Select a Recovery Point to failover to
 - **Latest**
 - Processes all data that's sent to Site Recovery service.
 - Creates a recovery point for each virtual machine
 - Used by VM during failover.
 - Lowest data loss (RPO, recovery point objective)
 - As VM created after failover has all the data that has been replicated to Site Recovery service when the failover was triggered.
 - **Latest processed**
 - Fails over all virtual machines of the recovery plan to the latest recovery point that has already been processed by Site Recovery service.

- If you are doing failover of a recovery plan, you can go to individual virtual machine and look at Latest Recovery Points tile to get this information.
- Low recovery time (RTO, recovery time objective)
 - As no time is spent to process the unprocessed data.
- **Latest app-consistent**
 - Fails over all virtual machines of the recovery plan to the latest application consistent recovery point that has already been processed by Site Recovery service.
 - When you are doing test failover of a virtual machine, time stamp of the latest app-consistent recovery point is also shown.
 - If you are doing failover of a recovery plan, you can go to individual virtual machine and look at Latest Recovery Points tile to get this information.
- **Latest multi-VM processed**
 - This option is only available for recovery plans that have at least one virtual machine with multi-VM consistency ON.
 - VMs that are part of a replication group failover to the latest common multi-VM consistent recovery point.
 - Other virtual machines failover to their latest processed recovery point.
- **Latest multi-VM app-consistent**
 - Only available for recovery plans that have at least one virtual machine with multi-VM consistency ON.
 - Virtual machines that are part of a replication group failover to the latest common multi-VM application-consistent recovery point.
 - Other virtual machines failover to their latest application-consistent recovery point.
- **Custom**
 - If you are doing test failover of a virtual machine, then you can use this option to failover to a particular recovery point.
 - !Only supported when failing over to Azure.

Azure Backup vs Azure Site Recovery

- Azure Backup is used to protect and restore data at a more granular level.
 - E.g. if some files become corrupted, you can use Azure Backup to restore them
- Azure Site Recovery is used to replicate the configuration and data of a system to another data center, then you would use.

Concept	Explanation	Backup	Disaster Recovery (DR)
Recovery point objective (RPO)	The amount of acceptable data loss if a recovery needs to be done.	Wide variability in their acceptable RPO. E.g. VM backups usually 1 day, DB backups 15 minutes.	Low RPOs. The DR copy can be behind by a few seconds or a few minutes.
Recovery time objective (RTO)	The amount of time that it takes to complete a recovery or restore.	Larger RPO -> the amount of data needed to process is much higher -> leads to longer RTOs. E.g. days to restore data from tapes, depending on the time it takes to transport the tape from an off-site location.	Smaller RTOs because they are more in sync with the source. Fewer changes need to be processed.
Retention	How long data needs to be stored	For scenarios that require operational recovery (e.g. data corruption, inadvertent file deletion, OS failure), backup data is typically retained for 30 days or less. 💡 From a compliance standpoint, data might need to be stored for months or even years. Backup data is ideally suited for archiving in such cases.	Needs only operational recovery data, which typically takes a few hours or up to a day. 💡 Because of the fine-grained data capture used in DR solutions, using DR data for long-term retention is not recommended.
Product	Use cases		
Azure Backup	• Accidental deletion • Patch Testing • Alternative location recovery • Security • Long-term data retention		
Azure Site Recovery	• Disaster Recovery with quick failover • Migration • Dev/Test Environment (e.g. test failover)		

3.7. Monitoring

Monitoring

Azure Network Watcher

- Inbuilt in Portal (*Monitor -> Network*)
- Features
 - **Topology**: e.g. VNETs, subnets, VMs, NICs
 - **Variable Packet Capture**: Captures TCP packages at NIC level as Wireshark files.
 - **IP Flow Verify**: Troubleshoots NSG
 - **Next hop**: Troubleshoots route tables
 - **Connection troubleshoot**: Why it does not connect?
 - Diagnostics Logging
 - Security Group View
 - NSG Flow Logging
 - VPN Gateway Troubleshooting
 - Network Subscription Limits
 - Role Based Access Control

Network Monitor

- Troubleshooting blade.
- Available for the following network resources:
 - ExpressRoute, VPN Gateway, Application Gateway, Network Security Logs, Routes, DNS, Load Balancer, and Traffic Manager.
- Available through Portal, CLI, PowerShell, Rest API, retrieved using Power BI or third-party logs.
- Events are logged in storage accounts, ready to be sent to Event Hub or Log Analytics.
- Metrics for performance measurements are collected over a period of time.

Azure Security Center

- Unified security management and advanced threat protection for workloads running in Azure, on-premises, and in other clouds.
- Enables you to discover and assess the security of your workloads and to identify and mitigate risk.

- E.g. • network recommendations • network maps • internet facing endpoints without NSGs.
- Two tiers: Free and Azure Defender

Azure Defender

- Formerly known as **Azure Security Center Standard Edition**
- Regulatory compliance dashboard and reports
- Threat protection for
 - Azure VMs and non-Azure servers
 - PaaS services
- Microsoft Defender for Endpoint (servers)

Just-in-time (JIT) virtual machine (VM) access

- Can be used to lock down inbound traffic to VMs
- Reduces exposure to attacks while providing easy access to connect to VMs when needed.
- Flow
 - i. A user requests access to a VM
 - ii. Azure Defender checks that the user has Role-Based Access Control (RBAC) permissions that permit them to successfully request access to a VM.
 - iii. Azure Defender automatically configures the Network Security Groups (NSGs) to allow inbound traffic to the selected ports and requested source IP addresses or ranges, for the amount of time that was specified.
 - iv. After the time has expired, Azure Defender restores the NSGs to their previous states.
 - Those connections that are already established are not interrupted

Azure Monitor

- Helps you monitor your applications and resources
- All data connected by Azure Monitor fits into two types: Logs, Metrics
- Log Analytics and Application Insights have been combined into Azure Monitor
 - Log analytics was a product before, now just referred as a feature.
- □ In 2018 "Operations Management Suite (OMS)" and [its services](#) were rebranded into Azure Monitor and it's no longer available.

Azure Logs

- Allows you to monitor log data
- Log data can be from on-premises or cloud resources
- Helps maintain resource availability and performance.
- It allows you to run queries and list or visualize results
- Reads data from a log analytics workspace

Log Analytics workspace

- Logical storage unit where log data is collected and stored
- Basic management unit of Azure Monitor Logs
- Retention: min 31 days (default), up to 730 days (paid)
- Supports sending custom logs through
 - Log Analytics Agent
 - [HTTP Data Collector API](#) (previously known as Log Analytics Data Collector API)

Queries in Log Analytics workspace

- Allows to query logs using KQL (Kusto Query Language)
- Allows you to save queries
- You can filter, sort, group query results or render charts based on them

Azure Metrics

- Feature of Azure Monitor that collects numeric data
- Save data into a time series database
- **Metrics**
 - Numerical values that are collected at regular intervals
 - Describe some aspect of a system at a particular time
 - Also called performance counters
- There is no need to set up additional diagnostics for metrics, nor opt-in for the data.
- Frequency of one minute.
- Can be viewed on portal using Azure metrics explorer
- Retention is often 93 days
 - !☐ But only lets you view 30 days at a time
 - Archive metrics to storage for longer retention
 - Can be configured in the settings for the resource.

- **Multi-dimensional metrics**
 - Some metrics can also name-value pair attributes (called dimensions)
 - Allows further segmentation
- Use-cases
 - Track the performance of your resource (such as a VM, website, or logic app) by plotting its metrics on a portal chart and pinning that chart to a dashboard.
 - Perform advanced analytics or reporting on performance or usage trends of your resource.
 - Get notified of an issue that impacts the performance of your resource when a metric crosses a certain threshold.
 - Archive the performance or health history of your resource for compliance or auditing purposes.
 - You can choose to stream this information to an event hub.
 - Doing so allows you to route them to Azure Stream Analytics for near-real-time analysis.

Agents

- Used for collecting data from compute resources and sending to Azure
- Helps measuring performance and availability of guest OS

Azure Monitor agent

- Also known as AMA
- Will eventually replace and consolidate Log Analytics Agent and Diagnostic extension
- Sends data to Azure Monitor Metrics or Log Analytics workspace

Log Analytics Agent

- Sends data to log analytics workspace
- Collects custom logs, IIS logs, performance counters, syslog, windows event logs.
- Can be installed on Azure VMs using the Azure Log Analytics VM extension for Windows and Linux
- For machines in a hybrid environment using setup, command line, or with Desired State Configuration (DSC) in Azure Automation.
- Outbound connection over TCP port 443

Log Analytics Gateway

- Formerly known as **OMS gateway**
- Allows sending logs from computers that are not connected to internet
- Agents send their data to gateway, gateway forwards it to log analytics workspace
- HTTP forward proxy that supports HTTP tunneling using the HTTP CONNECT command
- 💡 If long sending machines have no connection to internet use Log Analytics Gateway
- !❑ Computer that has gateway installed required Internet access as Private Link (ExpressRoute cables to Azure) is not supported for it (see [supported services](#))

Azure Diagnostics extension

- Collection of diagnostic data on a deployed application
- Supports
 - Azure cloud service (classic) Web and Worker roles
 - Virtual machines
 - Virtual machine scale sets
 - Service fabric
- Can send data to
 - Azure Monitor Metrics
 - Event hubs (for sending data outside Azure)
 - Azure Blob Storage
 - Application Insights
- Collects: • Performance counter metrics • Application logs • Windows Event logs • .NET EventSource logs • IIS Logs • Manifest based ETW logs • Crash dumps (logs) • Custom error logs • Azure Diagnostic infrastructure logs.
- Data storage
 - The extension stores its data in an Azure Storage account that you specify.
 - You can also send it to Application Insights.
 - Another option is to stream it to Event Hub, which then allows you to send it to non-Azure monitoring services.
 - You also have the choice of sending your data to Azure Monitor metrics time-series database

Application Insights

- Application Performance Management (APM) service for web developers building and managing apps on multiple platforms.
- Use it to monitor your live web application.

- It will automatically detect performance anomalies.
- It integrates with your DevOps process, and has connection points to a variety of development tools.
- It can monitor and analyze telemetry from mobile apps by integrating with Visual Studio App Center and HockeyApp.
- Saves data either in
 - Classic (application insights workspace)
 - Or in log analytics workspace

Azure Advisor

- It analyzes your resource configuration and usage telemetry. It then recommends solutions to help improve the performance, security, and high availability of your resources while looking for opportunities to reduce your overall Azure spend.
- Across subscriptions
- You can apply filters to display recommendations for specific subscriptions and resource types.
- The recommendations are divided into four categories:
 - **High Availability** : To ensure and improve the continuity of your business-critical applications.
 - **Security** : To detect threats and vulnerabilities that might lead to security breaches.
 - **Performance** : To improve the speed of your applications.
 - **Cost** : To optimize and reduce your overall Azure spending.

Azure Service Health

- Provides personalized guidance and support when issues in Azure services affect you
- Helps you prepare for upcoming planned maintenance.
- Alerts you and your teams via targeted and flexible notifications.
- Service Health tracks three types of health events:
 - **Service issues**
 - Problems in the Azure services that affects customers right now.
 - **Planned maintenance**
 - Upcoming maintenance that can affect the availability of services in the future
 - **Health advisories**
 - Changes in Azure services that require attention
 - E.g. • Azure features are deprecated • You exceed a usage quota

- Data is presented at
 - Azure Portal: Personalized Service Health Dashboard
 - Azure Monitor: Service Health Alerts
 - <http://status.azure.com> : General Health Overview of All Azure Services

Power BI

- Turn data processing to analytics and reports that provide real-time insights into your business.
- Works with cloud-based or on-premises data.
- Has a multitude of Azure connections available.
- Shape and refine your data to build customized reports.
- Products: • Power BI Desktop • Power BI Pro • Power BI Premium • Power BI Mobile • Power BI Embedded • Power BI Report Server
- You can use data sources from Azure e.g. Azure SQL Database, Azure HDInsight, Azure Blob Storage and many more.

3.7.1. Azure Sentinel

Azure Sentinel

- Solution for
 - security information event management (SIEM)
 - security orchestration automated response (SOAR)
- Delivers
 - intelligent security analytics
 - threat intelligence
- Key benefits
 - Speed
 - Scalability
 - AI/automation to improve effectiveness
 - Ability to consume data from different sources
- Provides solutions for security operations such as
 - **Collecting**
 - Collect data, analyze and parse it to some common format and place it in Log Analytics workspace
 - Sources include on-premises and multiple cloud

- Can be across users, devices, applications
- **Detecting** - alert detection
 - Minimize false positives by using an unparalleled threat intelligence
 - Detects previously undetected threats using machine learning
- **Investigating** - threat visibility and proactive hunting
 - Enhances with artificial intelligence
 - Allows hunting for suspicious activities
- **Responding** - threat response
 - Built-in orchestration and automation of common tasks.
- Integrates with
 - Log Analytics as data storage
 - Logic Apps to e.g. automate security responses, see playbooks
- It's resource is called "Azure sentinel workspace"
- You can see incidents from multiple Azure Sentinel workspace or tenants
 - Azure Lighthouse allows you to manage multiple tenants
 - You can hunt in different workspaces in same query

Workbooks

- Interactive dashboards
- There are gallery of workbooks
 - E.g. nice graphs for insecure protocols (LDAP, SMB, Kerberos...)
- You can also create your own workbooks using queries
- Can be exported/imported as JSON file, called Gallery Template
- 👁️ See [Workbooks | Azure/Azure-Sentinel \(GitHub\)](#) for workbook examples

Analytics rules

- Also known as **detection rules**
- Can be
 - built-in: there are already many
 - or custom: with custom triggers, periodicity etc.
- Each rule triggers an alert based on a condition
 - When an alert is generated it can run playbook
 - or *automation rules* that can tag, assign, or run a playbook
- Can enable User and Entity Behavior Analytics (UEBA)
 - Can map query results to entities e.g. a malware or a security group
- Optionally creates incidents

- Each rule can have tactics
 - Used to help with filtering rules and classification
 - [MITRE ATT&CK](#) framework
- Built-in rules has different alert types
- Uses Kusto Query Language (KQL) as query language
- 📖 See [Azure/Azure-Sentinel | GitHub](#) for query examples

Rules alert types

- **Microsoft security** (can be custom)
 - Creates incidents every time an alert is triggered in a connected Microsoft security solution
 - E.g. Microsoft Cloud App Security, Microsoft Defender for Identity
- **Fusion** (built-in only)
 - Uses machine learning to correlate low-fidelity events
 - E.g. "mass file download following suspicious Azure AD sign-in", [see more](#)
- **Machine learning behavioral analytics** (built-in only)
 - Based on proprietary Microsoft machine learning algorithms
- **Scheduled** (can be custom)
 - Based on built-in queries written by Microsoft security experts
 - E.g. "Malware in the recycle bin"
- 📖 Read more: [Detect threats out-of-the-box | Microsoft Docs](#)

Automatic URL detonation

- Enrich alerts with screenshots (e.g. for phishing sites), verdicts, final URLs
- Query results can map to a new URL entity type
- Can configure URL entities in analytics rules

User and Entity Behavior Analytics (UEBA)

- Detect any anomalous behavior of users and entities by profiling them
- Entities include non-user accounts e.g. computers, services etc.
- Can detect e.g. • abuse of privileged identities • compromised user and entities • insider threats • data exfiltration
- E.g. if user downloads 10 MB per day and downloads many GBs it would trigger an alert
- Data sources include: • Audit logs • Azure activity • Security events • Signing logs
- Used by both Azure Sentinel and Defender for Identity"

- Enabled and data is stored inside Sentinel workspace
 - No data is saved in Log Analytics workspace

Hunting

Queries

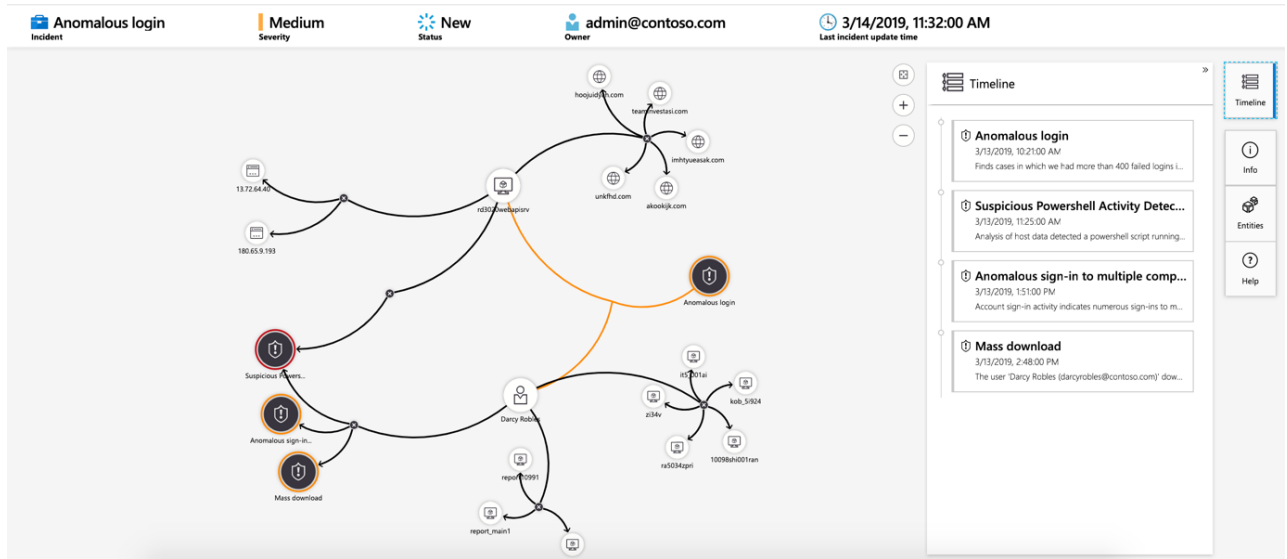
- Run built-in queries or own queries with KQL
- You can save queries to run later
- You can add tags
- **Azure Sentinel bookmarks**
 - You can tag and bookmark query results
 - Allows you to investigate using investigation map
- **Azure Sentinel livestreams**
 - Allows you to
 - test queries without conflicting with existing rules
 - get notified when it hits without creating a rule rules
 - add queries to livestreams to run them later or in a scheduled manner.
 - investigate them
 - 💡 Good to start with creating livestreams and if queries work promote them to creating rules.

Azure Sentinel Notebooks

- Integrates Jupyter notebooks for hunting using an indeterminate component
 - Charged for notebook compute + storage
- Can also save data as HTML/JSON
- !❑ Requires Azure Machine Learning workspace
- 📖 See [GitHub page \(Azure/Azure-Sentinel-Notebooks\)](#) for open-source library of samples

Investigation graph

- Also known as **investigation map**
- Allows you to investigate entity relationships in incidents
- Displays entity relationships extracted automatically from the raw data
- Requires using "entity mapping fields" in analytics rules



Playbooks

- Helps to automate and integrate across tools
- Can be triggered from an alert or incident investigation
- Can be used for e.g.
 - incident management: to open a ticket in JIRA/service now
 - enrich investigation: • GeoIP lookups
 - remediation: • block IP address/user access • isolate machine • trigger conditional access
- 📄 See [GitHub page](#) for open-source library of samples
- Done using Logic apps + APIs

Incidents

- Container for related alerts
- You can
 - assign incident to someone
 - change its severity
 - track and change its status (new, done etc.)
 - add bookmarks, tags and comments
 - investigate in "Defender for Endpoint"
 - Shows data connected to Sentinel
 - Also collects data in Log Analytics Workspace
 - can execute playbooks for automating stuff like

- integrating with a ticketing system using logic app
- automatically assigning an incident to someone when it's created
- can be investigated using investigation graphs

Machine learning

- Can use built-in models or bring your own models
- E.g. [anomalous RDP detection](#) for unusual IP/geolocation or new user
- Calculates possible kill chain
- Options
 - i. **Azure Machine Learning**
 - Run models hosted in the Azure Sentinel Notebooks
 - 💡 Easier, good for small data sets
 - ii. **Azure Databricks/Apache Spark**
 - You can
 - bring your own data via EventHub or Azure Blobs
 - or export the data from Azure Sentinel Log Analytics tables
 - 💡 Good for deploying and operating models for larger data

Bring Your Own Machine Learning (BYO-ML) platform

- Used for using own machine learning models
- Works with both Azure Databricks/Apache Spark and Jupyter Notebooks options
- Includes, samples, sample data, templates and libraries to communicate with Log Analytics (LA)
- Read more: [Bring your own ML | Microsoft Docs](#)

Azure Sentinel Fusion

- Helps reduction of noise by preventing alert fatigue
- Bring probabilistic kill chain to find novel attacks using machine learning
- !❑ Does not work without data connectors:
 - Azure Active Directory Identity Protection
 - Microsoft Cloud App Security
- 🗉 Read more: [Reducing security alert fatigue using machine learning in Azure Sentinel | Azure blog](#)

Data sources

- **Azure services**
 - Azure AD, Activity Logs, AIP, ASC, AzWAF...
 - Microsoft 365 Defender, Azure Defender, Microsoft 365 sources
- **3rd parties**
 - Cloud platforms such as AWS.
 - Others e.g. Symantec, Cisco, Citrix...
 - **Threat intelligence**
 - Lets you import the threat indicators from 3rd parties
 - Imports log events that can be used in queries, notebooks, workbooks and rules.
 - Providers include MISP, Palo Alto, Check Point...
 - Via Microsoft Graph Security API
 - Microsoft Host Integration Server (HIS) can be used to integrate IBM solutions.
- **Custom**
 - Azure Sentinel receives custom data over HTTPs (443)
 - Supports • REST-API • Common Event Format (CEF) • Syslog over port 443
 - Other protocols (e.g. syslog 514) should use middleware (e.g. a linux agent) to transform data to log analytics REST HTTPs
 - **Log Analytics agent**
 - Can be installed on physical and Windows/Linux virtual machines
 - E.g. can be installed on a Log Analytics server
 - **!** Connector proxy can be deployed if all machines should not be open to Internet
 - E.g. Log Analytics gateway for log analytics agents

Watchlists

- Custom CSV data you can upload to Azure
- Its data can be used for correlation with the events in Sentinel
- Can be used in search queries, detection rules, threat hunting, and response playbooks
- E.g. using KQL:
 - `let watchlist = (_GetWatchlist('CustomWatchListName') | project CustomColumnName);`
 - `Heartbeat`
 - `| where ComputerIP in (watchlist)`

Pricing

- Pay for per gigabyte (GB) for the volume of data ingested for analysis
- Can purchase **Capacity Reservations** for CapEx commitment, will be up to 60% cheaper.
- Main costs
 - Sentinel ingestion
 - Log Analytics ingestion
 - Storage (retention)
- No charges for queries
- Other (optional) costs for other integrated systems e.g.
 - Azure Logic Apps activations
 - Azure Notebooks (Jupyter hunting books)
 - BYO Machine Learning
 - Extract data from tenant
 - Additional retention for Log Analytics (after >90 days)
- 📖 See [its pricing page](#) and [Azure Pricing calculator](#)

Onboarding Azure Sentinel

- Prerequisites
 - Log analytics workspace
- Steps
 - Enable Azure Sentinel
 - Done by connecting Azure Sentinel into an existing Log analytics workspace
 - **!** Once enabled, workspace cannot be moved to other resource groups or subscriptions.
 - Connect your data sources
 - Steps: Main menu -> Data connectors -> Open connector page
 - Set up threat detection rules
- 📖 See also: [Quickstart: On-board Azure Sentinel | Microsoft Docs](#)

Log Analytics Workspace for Azure Sentinel

Retention

- Saved for 90 days by default

- !☐ Max available is 2 years (charged more)
- 💡 For long term storage, export data to storage account
- You can use Table Level retention for different retention settings based on data

One vs multiple log analytics workspaces

- 💡 Use 1 workspace if you can; both for Azure Security Center and Azure Sentinel
 - But can use multiple workspaces for e.g. granular access control, regulatory compliance.
- !☐ You can only connect 1 log analytics workspace at a time

Audit logs

- Can log who runs queries and query text
- Diagnostic settings -> Audit -> Select workspace / Event Hub / Storage to store logs
- If workspace is chosen, information is saved in a table called LAQueryLogs in workspace

Built-in RBAC roles

Role	Create and run playbooks	Create and edit dashboards, analytic rules, and other Azure Sentinel resources	Manage incidents (dismiss, assign, etc.)	View data, incidents, dashboards and other Azure Sentinel resources
Azure Sentinel Reader	-	-	-	✓☐
Azure Sentinel Responder	-	-	✓☐	✓☐
Azure Sentinel Contributor	-	✓☐	✓☐	✓☐
Azure Sentinel Contributor +	✓☐	✓☐	✓☐	✓☐

Role	Create and run playbooks	Create and edit dashboards, analytic rules, and other Azure Sentinel resources	Manage incidents (dismiss, assign, etc.)	View data, incidents, dashboards and other Azure Sentinel resources
Logic App Contributor				

4.1. Azure Resource Manager


Azure Resource Manager

- Designed to represent each service in Azure as a resource provider and each service instance in Azure as a modular resource.
- JSON templates are used to deploy collections of resources using Infrastructure-as-Code concepts.
- You can interact with Resource Manager using PowerShell, CLI, Client libraries, Visual Studio, Portal, REST API.

Resource groups

- Common lifecycle for resources: They can be created, managed, monitored, or deleted together.
- The Resource Manager also offers the concept of resource group templates
 - You define a service unit in advance, and then use the template to create as many resource groups as you need.

Azure Resource Manager (ARM) Objects

-  Envision your solution using ARM
 - Start by designing and conceptualizing your entire solution considering all components that may compose your solution.
 - Then identify individual units of functionality and find resources available on Azure that can facilitate the specific functionalities.
- **Resource** : Single service. E.g. web app, app service plan, SQL database.
- **Resource group** : Logical grouping of resources.

- **Resource group template** : JSON file that describes a set of resources.

ARM Templates

- Some or all of the properties of the resource can be parameterized so that you can customize your deployment by providing parameter values at deployment time.
- Deployment
 - ARM Templates are deployed in a few ways.
 - These depend on your aims, the result intended and your chosen method for development.
 - A **developer** may choose to use Visual Studio to create and deploy ARM templates directly and to manage the lifecycle of the resources through Visual Studio.
 - An **administrator** may choose to use PowerShell or the Azure Command Line to deploy resources and amend them.
 - An **end user** without command line or developer skills would choose to use the Azure Portal to deploy resources without realizing a template is involved. E.g. marketplace offerings.
- Advantages:
 - **Ensure idempotency** : Identical template to multiple resource => same functionality.
 - **Simplify orchestration** : Automate.
 - ***Configure multiple resources** : Order, fix dependencies.
 - **Parameterize** : Define input & input for reuse. Can be nested for larger orchestration.
- Template resources: Parameters (=> *Variables*) => Resources => Output

JSON schema

-  Empty ARM template:
- ```
{
 "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
 "contentVersion": "1.0.0.0",
 "parameters": { },
 "variables": { },
 "resources": [],
 "outputs": { }
}
```
- Required: content, resources
- Optional: parameters, variables, output.

- Sources:
  - The Azure Quickstart templates on Github are
  - Inspect Automation script element of objects.

#### 4.1.1. Role-Based Access Control (RBAC)

### Role-Based Access Control (RBAC)

- You can assign roles to existing Azure AD identities that grants them pre-determined levels of access to an Azure subscription, resource group or individual resource.
- Some built-in roles:
  - **Owner** : Owner can manage everything, including access.
  - **Contributor** : Contributors can manage everything except access.
  - **Reader** : Readers can view everything, but can't make changes.
  - **User Access Administrator** : Allows you to manage user access to Azure resources.
  - **Virtual Machine Contributor** : Allows you to manage virtual machines, but not access to them, and not the virtual network or storage account they are connected to.

## Role Assignment

- Associates a **[security principal]**(#security-principals) to a **role** in a **given scope**.

### Security principals

- **Users**
  - Users in AD of the subscription.
  - Can be assigned to external Microsoft accounts in same directory.
- **Groups**
  - AD security groups.
  - Best practice.
- **Service principals**
  - Service identities.
    - Authenticates with Azure AD to communicate with each other.
  - Can be granted access to other resources by assigning roles.



## Resource Scopes

- Subscriptions, resource group, individual resources.
- Resource inherits assignments from its parent resources.
  - Access inheritance: Subscription => Resource Groups => Resources
- **Scoping to Resource Groups**
  - Add/remove and modify resources quickly without having to recreate assignments and scopes
  - Owner or contributor access => Does not require additional administrator assistance or having access to resources in other resource groups.

## Custom roles

- Use REST API.
- **!** Azure AD tenant is limited to 2000 custom roles.
- Steps:
  - Create a role definition with assignable scopes.
  - Assign the role definition to a scope.

## Creating a new role definition

- 🛠️ To create a new custom role you run the `New-AzureRmRoleDefinition` cmdlet
- You can pass a JSON template to the cmdlet or use `PSRoleDefinitionObject`.
- E.g. json:

```
{
 "Name": "New Role 1",
 "Id": null,
 "IsCustom": true,
 "Description": "Allows for read access to Azure storage and compute resources",
 "Actions": [
 "Microsoft.Compute/*/read",
 "Microsoft.Storage/*/read",
],
 "NotActions": [
],
 "AssignableScopes": [
 "/subscriptions/c489345-9cd4-44c9-99a7-4gh6575315336g"
]
}
```

### 4.1.2. Azure Resource Policies

## Azure Resource Policies

- Resource Policy is a service that controls which resources can be created and what for they take, such as naming conventions, locations, and sizes.
- Ensure that resources created in the cloud comply with corporate standards and service level agreements.
  - E.g. preventing users from creating resources in specific locations and of specific types and sizes
- Contains two elements, a policy definition, and a policy assignment
- There are many pre-defined policy definitions built into Azure.

## Policy vs. RBAC

- RBAC controls user access, permissions, privileges, and actions at different scopes.
- Policy evaluates resource properties for already existing resources and during deployment
  - E.g. control which type of resource is available for deployment in your organization
  - E.g. limit the locations in which you deploy resources or require your resources to follow naming conventions
- **Azure policy** is a default allow, and explicit deny system as opposed to RBAC.

## Creating policies

- The contributor role does not contain the necessary permissions.
- You need:
  - **Microsoft.Authorization/policydefinitions/** write permission to define a policy.
  - **Microsoft.Authorization/policyassignments/** write permission to assign a policy.

## Built-in Policies

- E.g. allowed locations, allowed resource types, allowed storage account SKUs, allowed virtual machine SKUs, apply tag and default value, enforce tag and value, not allowed resource types, require SQL Server version 12.0, require storage account encryption.
  - Inclusion is to limit the number a user is required to create to manage their subscription efficiently.

## Policy Definition

- Every policy definition contains a minimum of two elements:
  - Conditions under which it is enforced.
  - An action that triggers if the conditions are met.
- Policy definition JSON elements: • mode • parameters • display name • description • policy rule • logical evaluation • effect
- E.g.
 

```

{
 "if": {
 "field": "type",
 "in": "[parameters('listOfResourceNotAllowed')]"
 },
 "then": {
 "effect": "Deny"
 }
}

```
- During assignment, the list of resource types not allowed parameter is populated and evaluated against all current resources in scope for the assignments.
- Any non-compliant resources will be identified. When creating new resources, the deployment will fail if they are non-compliant.
- The definition can contain *multiple if, then* blocks and combines logical operators, conditions, and fields to build out the policy.
- The policy can define alternative effects.
  - Alternate effects: • Deny • Audit • Append • AuditIfNotExists • DeployIfNotExists
- You can assign any of these policy definitions through the Azure portal, PowerShell, or Azure CLI.

## Policy Assignment

### Scope

- Occurs over a specific scope.

- A scope is the list of all the resource groups, subscriptions, or management groups that the policy definition is assigned to.
- Policy assignments are inherited by all child resources.
  - This can be altered or prevented using the exclusion option at the time of assignment.
- The scope, exclusions, and parameters of the NotAllowed resource types are shown in the policy assignments blade when assigning a definition.

## Initiative Definition

- Collection of policy definitions that are designed to achieve a design goal.
- Can be assigned as a group to simplify the process of achieving that goal.
- Appears as a single item.

## Policies for Naming Conventions

- These can use wildcards, patterns tags, and multiple patterns to apply restrictions to your Azure resource names.
- E.g. a pattern match asserting that is the resource name does not begin *contoso* and have six characters as the rest of the name then it will be non-compliant.
  - E.g. the resource name must begin *contoso* and have six characters as the rest of the name.

```

◦ {
◦ "if": {
◦ "not": {
◦ "field": "name",
◦ "match": "contoso??????"
◦ }
◦ },
◦ "then": {
◦ "effect": "deny"
◦ }
◦ }

```

## Management groups

- Groups multiple subscriptions within an enterprise.
- Allows you to enforce policies on multiple subscriptions.

### 4.1.3. Securing ARM templates (Azure Key Vault)

## Securing ARM templates

## Azure Key Vault

- Create, manage and import **secrets**, **keys**, and **certificates** for applications, services and users.
- When deploying resources using Arm templates and automating that deployment, it is best practice to use a **Service Principal**.
  - In on-prem AD it was called: Active Directory Service Account
- The premium tier allows storage of these secrets in a Hardware Security Module, a physical device to contain all your secrets.
- Flow:
  - Administrator creates & manages vaults and keys.
    - Can be created by any contributor/owner.
  - Sends URIs to developers.
  - Security administrators uses usage logging for keys.
- Dev/test keys can be migrated to production use at deployment.
  - **Key Vault Use in ARM Templates**
    - Embedding credentials and passwords inside a template are unwise.
    - To further secure the deployment, it is advised to create an Azure Service Principal.
    - With key vaults the value is never exposed because you only reference its key vault ID.
    - Use in ARM templates
      - a. Set the `enabledForTemplateDeployment` property to true when you create the Key Vault.
      - b. Create secret to be used in template
      - c. Ensure template can access Key vault
        - Ensure the service principal, user or template has the **Microsoft.KeyVault/vaults/deploy/action** permission for the correct Key Vault
        - The Contributor built-in role already has this permission.
      - d. Reference the secret using a static ID in the template parameter file.
        - **! Challenge:** Sometimes the Key Vault secret will change from deployment to deployment
          - It this requires the use of a dynamic ID reference.
          - It cannot go in the parameter file.

- 💡 Solution: Nested template where key is also deployed.

#### 4.1.4. Deploying ARM templates (Azure Building Blocks)

### Deploying ARM templates

## Problem definition

- Authoring and deployment of ARM templates can be hard:
  - Become very complex in a short amount of time.
  - Difficult to ensure that Microsoft best practices for Azure infrastructure are reflected in every template authored by your team.
- Development of your templates follows a pattern:
  - E.g.:
    - a. JSON object for a virtual network
    - b. JSON object for virtual machines that are deployed into the virtual network.
    - c. JSON object for a network security group to secure the virtual network.
    - d. JSON object for a load balancer to distribute incoming requests to the virtual machines
- Problem:
  - Great deal of knowledge about Azure Resource Manager and the resources themselves.
  - Difficulty maintaining your templates because any modification can lead to unforeseen issues.

## Azure Building Blocks

- Command line tool and set of ARM templates.
  - Reflect best practices as prescribed by the *Patterns & Practices team* at Microsoft.
- Designed to simplify deployment of Azure resources.
- Flow:
  - i. Specify settings for Azure resources using the Building Blocks JSON schema
    - You can either specify your resources settings in one large file or several small files.
    - Supports: • Virtual Networks • Virtual Machines (including load balancers) • Virtual Machine Extensions • Route Tables • Network Security Groups • Virtual Network Gateways • Virtual Network Connection

- ii. Command line tool merges these settings with best practice defaults (from GitHub templates) to produce a set of parameter files
- iii. The command line tool deploys these parameter files using a set of pre-built Azure Resource Manager templates

## Deploying resources using building blocks

1. Install **azbb\*** command line tool and the Azure CLI.

### 2. Create building blocks

- Create a JSON file with a `buildingBlocks` parameter containing an array of resources that you wish to deploy.
  - Each resource within that JSON array can have certain options configured.
  - E.g. Virtual Network resource => `addressPrefix_`, `subnets_` and `name`.
  - Any options you do not specify are set using the default options specified in the building blocks repository.
  - E.g. deploying a Virtual Network includes following properties:
    - `type` : identify the type of building block. E.g. `VirtualNetwork`.
    - `name` : Unique name of the resource
    - `addressPrefixes` : Array of address ranges in CIDR notation.
      - E.g.: `"addressPrefixes": [ "10.0.0.0/16", "11.0.0.0/16" ]`
    - `subnets` : an array—we can specify up to 1,000 subnets for each VNet.

### 3. Create a settings file

- Run the **azbb** command line tool to parse your settings and combine it with the default settings from the building blocks repository
  - The **azbb** command line tool will output an ARM template parameters file,
- Empty settings file:
  - {
  - `"$schema": "https://raw.githubusercontent.com/mspnp/template-building-blocks/master/schemas/buildingBlocks.json",`
  - `"contentVersion": "1.0.0.0",`
  - `"parameters" : { "buildingBlocks": { "value": [ {} ] } }`
  - }
- E.g. deploying a Virtual Network
- {

```

○ "$schema": "https://raw.githubusercontent.com/mspnp/template-
building-blocks/master/schemas/buildingBlocks.json",
○ "contentVersion": "1.0.0.0",
○ "parameters": {
○ "buildingBlocks": {
○ "value": [
○ {
○ "type": "VirtualNetwork",
○ "settings": [
○ {
○ "name": "msft-hub-vnet",
○ "addressPrefixes": [
○ "10.0.0.0/16"
○],
○ "subnets": [
○ {
○ "name": "firewall",
○ "addressPrefix": "10.0.1.0/24"
○ }
○]
○ }
○]
○ }
○]
○ }
○ }
○ }

```

#### 4. Deploy

- **azbb** tool will use the Azure CLI (version 2.0) to deploy the master ARM template from the building blocks repository using the parameters file that was generated.
  - The Azure deployment will create a collection of resources within an Azure Resource Group.
    - It'll have historical deployment information
      - You can go back and view the template and parameters used by the building blocks command line tool to deploy your resources.
- Command:
  - `azbb -g <new or existing resource group> -s <subscription ID> -l <region> -p <path to your settings file> --deploy`

#### 4.2. Managed Server Applications (App Service Environments, Azure Service Fabric, Azure Container Service, Azure Container Instance)



## Managed Server Applications in Azure

- Infrastructure-Backed Platform-as-a-Service (IaaS) architecture that can be used when automatic management of infrastructure is required.
- Every Microsoft Azure PaaS service is already hosted on Azure IaaS.
- Three services available:
  - i. **App Service Environments** provides a dedicated scalable home for Azure Web Apps.
  - ii. **Azure Service Fabric** provides a cluster of VMs to host containers and microservices for those applications.
  - iii. **Azure Container Service** provides open source tools to orchestrate containers based on Docker Swarm, Mesosphere and Kubernetes clusters

## App Service Environments

- App Services separate many of the hosting and management concerns for your web application.
- You want sometimes more control:
  - E.g. make sure that all of the virtual machines hosting your web applications do not allow any outbound requests. (PCI compliance)
- Allows you to configure network access and isolation for your applications.
- Allows you to scale using pools of instances far beyond the limits of a regular App Service plan instance
- Instances are dedicated to your application alone.
- Gives you more control but retain: **automatic scaling**, **instance management**, and **load balancing**
- Same concepts & paradigms as Virtual Machines:
  - Environments are created within a subnet in an Azure Virtual Network
  - You can use Network Security Groups to restrict network communications to the subnet where the Environment resides.
  - You can also use a protected connection to connect your Virtual Network to corporate resources
- Versions
  - **V1**: Supports both classic and Resource Manager Virtual Networks
  - **V2**: Only resource Manager resources.
    - Automates most of the scaling, creation, and maintenance which v1 requires to be carried out manually.

## Azure Service Fabric

- Solution making it simple to package, deploy, and manage scalable and reliable containers and microservices.
- Assists with the developing and managing cloud native applications.
- Intended for scalable container-based applications.
- Microservice based applications
  - Creates a cluster of VMs that runs containers per stateful or stateless microservice.
  - Assists with provisioning, deploying, monitoring and managing in fast and efficient, automated fashion.
  - Powers Service many Microsoft services today, including *Azure SQL Database, Azure Cosmos DB, Cortana, Microsoft Power BI, Microsoft Intune, Azure Event Hubs, Azure IoT Hub, Dynamics 365, Skype for Business* and many core Azure services.

## Application lifecycle management

- Supports the application lifecycle and CI/CD of cloud applications including containers.
- Simple workflows to provision, deploy, patch, and monitor applications
- Integrating with CI/CD tools such as Visual Studio Team Services, Jenkins, and Octopus Deploy.
- It hosts microservices inside containers that are deployed and activated across the Service Fabric cluster.”
  - **Service fabric cluster**
    - Node type count: 1 to 3
      - Per node, durability tiers: Bronze, Silver, Gold.

## Service Fabric Analytics

- Requires Azure Diagnostics
- Can be added from marketplace
  - Get insight into your Service Fabric framework
  - Get insight into the performance of your Service Fabric applications and micro-services
  - View events from your applications and micro-services
  - E.g. `StatefulRunAsyncCancellation`, `StatefulRunAsyncFailure`

## Azure Container Service

- Simplifies processes associated with creating, configuring, and managing virtual machines.
- Supports three orchestration services:
  - i. Docker Swarm
  - ii. Mesosphere DC/OS
  - iii. Kubernetes
- You are only charged for these VMs, Storage and networking resources actually used.
- Contains Azure Kubernetes Service

## Azure Kubernetes Service

- Azure manages health, maintenance, and monitoring.
- Easy scaling, automatic version upgrades, and self-healing master nodes.
- Can be deployed into VNet
- **Application routing:** get a public DNS endpoint Azure DNS that has direct ingress into Kubernetes cluster into the container that's providing the front-end of that application.
- **Container insights:** Logs/metrics from containers
- Access is provided to the Kubernetes API endpoints.
  - There is no SSH access to the AKS cluster.
- Orchestration:
  - Multiple copies of each containers for high availability and scalability.
  - Allows deploying container images across a set of VMs.
  - Configures networking
  - Service discovery between them.
  - Doing health monitoring on the containers.
- **Virtual Kubelet:** Microsoft project that allows Azure Container Instances to run on AKS.

## Azure Container Instance

- Only pay while it's building
- Do not pay when it's done

### 4.2.1. High-Performance Compute (HPC)

## High-Performance Compute (HPC)

- Typically describes the aggregation of complex processes across many different machines thereby maximizing the computing power of all of the machines.
- Used for massively scalable parallel processing of memory-intensive workloads
  - e.g. 3D rendering and tasks that process, transform, and analyze large volumes of data.
- For other cloud-based and even hybrid solutions, Azure provides access to both Windows and Linux HPC clusters.
- Through HPC in the cloud, one could create enough compute instances to create a model or perform a calculation and then destroy the instances immediately afterward.
- Advancements in the HPC allows machines to share memory or communicate with each other in a low latency manner.
- Features of Azure that support HPC
  - HPC Pack
  - Azure Batch for short-term massively scalable infrastructure
  - Stateless Component Workloads.

## Remote Direct Memory Access

- Remote Direct Memory Access, or RDMA, is a technology that provides a low-latency network connection between processing running on two servers, or virtual machines in Azure.
- From a developer perspective, RDMA is implemented in a way to make it seem that the machines are "sharing memory."
- RDMA is efficient because it copies data from the network adapter directly to memory and avoids wasting CPU cycles.
- VM sizes A8 to A19 => Most efficient way to run

## HPC Pack

- Microsoft's HPC cluster and job management solution for Windows.
- Can be installed on a server that functions as the **head node**
  - The server can be used to manage compute nodes in an HPC cluster.
- Can be in hybrid scenarios where you want to "burst to Azure" with A8 or A9 instances to obtain more processing power.
- Supports several Linux distributions to run on compute nodes deployed in Azure VMs, managed by a Windows Server head node.

## Azure Batch

- Managed HPC pack offering on Azure.
- Provides
  - Auto-scaling
  - Job scheduling
  - compute resource management
    - VMs of compute nodes
- Includes end-of-cycle processing such as a bank's daily risk reporting or a payroll that must be done on schedule.
- Includes large-scale business, science, and engineering applications that typically need the tools and resources of a compute cluster or grid.
- Batch works well with **intrinsically parallel** (sometimes called "**embarrassingly parallel**") applications or workloads, which lend themselves to running as parallel tasks on multiple computers.

## Concepts


- **Pool**
  - Number and size of machine
  - E.g. 100 machines, n1 series.
- **Job**
  - Includes one or more code packages.
- **Task**
  - Task to execute code package.
- **Execution:**
  - i. Starts all VMs (takes time)
  - ii. Executes code
  - iii. Removes all VMs

## Scaling out Parallel Workloads

- Through the **Batch API**
- Can be managed as part of a larger workflow managed by tools such as **Azure Data Factory**.
- You can wrap an existing application, so it runs as a service on a pool of compute nodes that Batch manages in the background.
  - You can develop the service to let users offload peak work to the cloud, or run their work entirely in the cloud.

- The Batch Apps framework handles the movement of input and output files, the splitting of jobs into tasks, job and task processing, and data persistence.

## Stateless component workloads

- You do not have to rely on the pre-defined HPC services.
- Azure provides non-managed solutions and related services to provide a degree of HPC using IaaS.
  - These include: • Virtual Machines • VM scale Sets • Azure Container Services • HDInsight • Machine Learning and more.
  - Azure fabric is a good platform to deploy parallel compute tasks on several services.
-  Best practices:
  - Azure Resource manager to automate deployment.
  - Auto scale based on various criteria.
    - E.g. deploying a scale set of VMs with auto-scale based on a custom metric with HPC Pack.

### 4.3. Migration strategies

#### Migration strategies

## On-premises lift and shift

- Benefits of pay as you go computing
- No need to rewrite application code to fit a cloud application pattern.

#### Cloud Maturity

1. **On-premises** monolithic architecture
2. **Lift and shift:** No re-architect, no code changes
  - Cloud infrastructure-ready monolithic architecture
  - Cloud DevOps ready monolithic architecture
3. **Architected for the cloud**, might require new code
  - Cloud optimized full PaaS & cloud-native with monolithic and microservices architectures.

## Azure Migrate services

- Discovery and assessment tool
- Assesses suitability for migration and ensures that sizing is correct for the performance of the VM
- Estimates of the cost of running an VM in Azure
- Visualize dependencies of a specific VM or for all VMs in a group
- **! Limitations**
  - Provides assessment for only VMWare VMs
    - If you want to assess Hyper-VMs and physical servers, use the **Azure Site Recovery Deployment Planner** for Hyper-V, and our partner tools for physical machines.
  - Only supports managed disks for migration assessment.
  - All regions are not supported

## Classic (Azure Service Manager) migration to ARM

- From Azure Service Manager (ASM) model to Azure Resource Manager (ARM) deployment.
- Supported services: • Virtual Machines • Availability Sets • Cloud Services • Storage Accounts • Virtual Networks • VPN Gateways • Express Route gateways • Network Security Groups • Route Tables • Reserved IPs

## Cloud to Platform-as-a-Service (PaaS)

- From cloud services to a PaaS solution
- Necessary to consider the difference between VMs, workloads, and applications in each model.

## Azure Cloud Services

- Platform as a service (PaaS) in Classic Model (not ARM)
- A cloud service deploys applications as VMs; code is connected to a VM instance which might be a Web role or a worker role
  - **Web role:** Automatically deploys and hosts your app through IIS.
  - **Worker role:** Does not use IIS, and runs your app standalone

*Scaling and management*

- To scale the application, more VMs are deployed.
- You don't create virtual machines. Instead, you provide a XML configuration file that tells Azure how many of each you'd like, such as "three web role instances" and "two worker role instances."
- You still choose what size those backing VMs should be.

*The deployment package*

- Contains the **web role** and **worker role** definition
- Specifies the instance count for each role; an instance is a VM hosting that role.

*Migrating a cloud service Service Fabric*

- Cloud Services with **Worker Roles** can be migrated to Service Fabric Cluster with Stateless Service
- Migrating a cloud service to Service Fabric switches to deploying applications to VMs that are already running Service Fabric either on Windows or Linux.
- The applications or services that are deployed are entirely unrelated to the VM infrastructure.
- The service fabric application platform hides the VM layer from the application.

*Handling dependencies*

- A cloud service application will typically have external dependencies.
  - E.g. services that manage the data and state of an application and the method of communicating between web and worker roles.
    - Such as Azure Redis, Storage Queue, Service Bus.
- A Service fabric application can also rely on the same external service dependencies.
- The quickest and easiest way to migrate a Cloud Service application to service fabric:
  - Convert the Web roles and worker roles to stateless services whilst keeping the architecture the same.
- If the aim is to remove the external dependencies and take full advantage of the ability to unify deployment, management and upgrade models, then state-full services would be required which will mean full code and application rewrites.

**4.4. App Services**



## App Services

### SKUs

- **Basic Tier**
  - Scaling is manual
- **Standard or upper service tiers**
  - Scaling is automatic
  - ! Standard tier allows to scale up to 10 instances.
    - ! If you still need more instances you can go to the **Isolated tier** where you can scale up to 100 instances
- ! Use **Standard** or **Premium** tiers in order to support autoscale and SSL.

### Authorization types

- **Allow all requests**
  - Use your own authentication and authorization code.
- **Allow only authenticated requests**
  - User is challenged or returned 401
- **Allow Anonymous requests**
  - Handles authentication & authorization
  - Defers authorization decisions to your application code

## App Types

### Web Apps

- PaaS offering to host web applications.
- Fully managed and easily configurable for e.g. such as AlwaysOn, custom domains, and autoscale.
- Supports .NET, Java, PHP, Node.js, or Python
- **Deploy:** Git, Kudu, Microsoft Visual Studio through FTP or Web Deploy protocol.
- **Autoscale**
  - Creates multiple instances of the Web App
  - Automatically load balanced to meet potentially demands

### Web app containers

- Linux variant can host docker containers directly using a Web App.
- Docker containers can be sourced from Docker Hub, Azure Container Registry or GitHub.
- Can be deployed manually, or deployed in a streamlined continuous integration process using Docker Hub or GitHub.

### API apps

- Specialized version of Web Apps.
- Support for developing, hosting and securing your custom APIs in the context of App.
- It can run either
  - custom code or
  - pre-built software to connect to existing popular SaaS solutions through Logic App.
- Integrates seamlessly with API Management.
- Easy authentication using service-to-service or CORS.



### Mobile Apps

- Mobile App endpoints are REST APIs.
- Provides capabilities of:
  - **Single sign on:** From list of Azure AD.
  - **Offline sync:** Work offline when connectivity is not available, and synchronize with your enterprise backend systems when devices comes back online.
    - Any data source including • SQL • Table Storage • Mongo • Document DB • any SaaS API including Office 365, Salesforce, Dynamics, or on-premises databases.
  - **Push notifications:** You can easily hook Notification Hubs to any existing app backend.
  - **Auto scaling**
- Client SDKs are available to connect mobile app to a Mobile App instance for its backend data.
  - Supported for: • Xamarin Android/iOS, • Android Native, • iOS Native, • Windows Store, • Windows Phone, • .NET, • HTML

## 4.5. Authoring Serverless Applications in Azure

# Authoring Serverless Applications in Azure

## Azure Functions

- Implemented using the existing code and base functionality for **Azure WebJobs**.
- Created using one of two models:
  - **App Service**
    - Functions exist as a kind of app within an App Service Plan.
  - **Consumption**
    - Allows you to pay for Function Apps based on execution time as opposed to having a dedicated App Service Plan.
    - Billed at a rate specific to Gigabyte Seconds after a specific free monthly allocation threshold has been met.
- Share a lot of functionality with other App types such as Web Apps.
  - E.g. • App Settings, • Connection Strings, • AlwaysOn • Continuous Deployment.
  -   Enable AlwaysOn for Function App to respond to requests as quickly as possible.

## Authorization

- **Anonymous:** No API key is required.
- **Function:** A function-specific API key is required.
- **Admin:** The master key is required.

## Triggers

- A **Trigger** is any event that invokes an Azure Function.
- A function can have triggers for many different scenarios including: • Creation of Blobs • Changes to data in Cosmos DB • External File or Table changes • HTTP requests • OneDrive or Excel files • E-mail messages • Mobile Apps • SendGrid e-mails • Twilio Text Messages
- A function can also be triggered by existing messaging services including: • Service Bus • Storage Queues • Logic Apps • Event Grid • Event Hubs • Webhooks

## Integration

## API Management


- Provides developer engagement, business insights, analytics, security and protection.
- **Flow**
  - Administrators create APIs.
    - Each API consists of one or more operations, and each API can be added to one or more products.
      - A **product** is segmentation/grouping of APIs.
        - Access policies can be enforced.
        - Defines APIs (endpoints)
  - **Subscribers**
    - Developers/B2B partners subscribe to a product that contains that API,
    - They call the APIs operation, subject to any usage policies that may be in effect.
- **Portals**
  - **Publisher portal:** Admin portal.
    - Configures APIs & products & subscribers.
  - **Developer portal:** Partners can test the services.
- **Example**

| API                                                                                                                                        | Products                                                                                                                                                      | Subscribers                                                                                 |
|--------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li>• DEVICEINFO • BUILDINFO • ORDERS • DELIVERYORDER • SHIPMENT • MILESTONES • RETURNDEVICE</li> </ul> | <ul style="list-style-type: none"> <li>• DCServicesExternal • DCServicesInternal • MakeInternal • MakeExternal • ReturnsInternal • ReturnsExternal</li> </ul> | <ul style="list-style-type: none"> <li>• Arvato • Ups • Fedex • Walmart • Amazon</li> </ul> |

## Logic Apps

- Workflows that can connect various components of your application together using minimal or no-code.
- Designed using JSON.
- Logic Apps can integrate with
  - Existing services using built-in connectors such as SQL Server • OneDrive • Dropbox • Twilio • SendGrid • Cosmos DB • Event Grid • Event Hubs • Storage Blobs, Queues and Tables • Mobile Apps
  - custom APIs that are deployed as API Apps in your subscription.
- The main components of a Logic App are as follows:
  - **Workflow:** The business process described as a series of steps, in an acyclic graph (loops are not supported).
  - **Triggers:** The step that starts a new workflow instance.
  - **Actions:** A step in a workflow, typically a Connector or a custom API App.
  - **Connector:** A special case of API App ready made to integrate a specific service (e.g., Twitter) or data source (e.g., SQL Server).
- E.g. Azure Service Bus Connector to queue processing that you will run on HDInsight against some big data, and then send the result via an SMS message using the Twilio connector.

## Traffic Manager

- Automatically fails over if the primary region becomes unavailable.
  - **Multi-region architecture / model**
    - Provides a higher availability of your applications.
    -  Have paired region as secondary region as paired regions are prioritized during outage.
- Supports different routing algorithms
  - E.g. priority routing (formerly called failover routing)
    - All requests are sent to prioritized endpoint until it's unhealthy.
- E.g. use SQL database geo-replication to the standby region.
  - **Active region:** Where application with SQL lies.
  - **Standby region:** Where application exists and SQL is replicated.

### 5.1. Application Architecture Patterns in Azure

## Application Architecture Patterns in Azure

- **Design patterns:** Repeatable best practice.
- **Microsoft Patterns & Practices**
  - Created & maintained by patterns & practices team at Microsoft
  - Available on [GitHub \(github.com/mspnp\)](https://github.com/mspnp)
- **Azure Architecture Center**
  - [Architecture Center Guide](#)

## Performance Patterns

### Modular Applications

- Makes it easier to design both current and future iterations of it.
- Existing modules can be extended, revised or replaced to iterate changes to full application.
- Modules can also be tested, distributed and otherwise verified in isolation.

### Stateless Applications

- A stateful application saves data about each client session and uses that data the next time the client makes a request.
- A stateless app is an application program that does not save client data generated in one session for use in the next session with that client.
- Advantages of stateless applications
  - Easy to scale horizontally
  - Can be cached easily
  - Less storage
  - No need to bind the client to the server as in each request the client provides all its information necessary

### The Valet Key Pattern

- **Problem**
  - Dependency on a storage mechanism is an overhead
  - Requires the client to have access to the security credentials for the store
  - After the client has a connection to the data store for direct access, the application can't act as the gatekeeper.

- It's no longer in control of the process and can't prevent subsequent uploads or downloads from the data store.
- Granular access: Lower performance, higher data transfer costs and the requirement to scale out.
- **Solution**
  - The Valet Key pattern
  - Mechanism where your application can validate the user's request without having to manage the actual download of the media.
  - Still keeps the media private and locked away from anonymous access.
  - E.g. SAS tokens in Azure Storage / EventHubs...
  - **Flow**
    - a. **Client** sends a request to the application to access a media asset
    - b. **Application** then validates the request
    - c. If the request is valid:
      - **Application** goes to the external storage mechanism
      - Generates a temporary access token
      - Provides following to the client:
        - the URI of the media asset to the **client**
        - Temporary access token.
  - **Result**
    - Up to the client application or browser to use the URI and temporary access token to download the media asset.
    - Storage service is responsible for scaling up to manage all the requests for media
    - Application focuses solely on other functionality.

## Command and Query Responsibility Segregation (CQRS) pattern

- Segregate read & update operations to data using separate interfaces.
- Maximizes performance, scalability, and security.
- Supports the evolution of the system over time through higher flexibility
- Prevents update commands from causing merge conflicts at the domain level.
- **Traditional CRUD designs**
  - Same entity, DTO (data transfer object) and DAL (data access layer) repository for read and write.
  - Complex domain may introduce:
    - Mismatch between the read and write representations of the data
      - E.g. additional columns or properties that must be updated correctly even though they aren't required as part of an operation.

- Update conflicts caused by concurrent updates when optimistic locking is used
- Security and permissions more complex because each entity is subject to both read and write operations, which might expose data in the wrong context.
- **CQRS**
  - Data models used for querying and updates are different.
    - The read model of a CQRS-based system provides materialized views of the data, typically as highly denormalized views, e.g. domain models.
  - Separation of the read and write allows each to be scaled appropriately to match the load.
    - E.g. reads encounter more load than writes.
  - When the query/read model contains denormalized data (see [Materialized View](#) pattern), performance is maximized when reading data for each of the views in an application or when querying the data in the system.

## Throttling pattern

- A strategy to autoscaling is to allow applications to use resources only up to a limit, and then throttle them when this limit is reached.
- Enables the system to continue functioning and meet any SLAs.
- The system should monitor how it's using resources so that, when usage exceeds the threshold, it can throttle requests from one or more users.

## Resiliency Patterns

### Transient Errors

- Errors that occur due to temporary interruptions in the service or due to excess latency.
- Many of these temporary issues are self-healing and can be resolved by exercising a retry policy.
- Handling transient errors: Big difference between on-premises and cloud applications.
- A break in the circuit must eventually be defined so that the retries are aborted if the error is determined to be of a serious nature and not just a temporary issue.

### *Transient Fault Handling*

- Managing connections and implementing a retry policy
- Implemented in many .NET libraries such as [Entity Framework](#) and [Azure SDK](#).



### *Circuit breaker pattern*

- If everyone is retrying due to a service failure, there could be so many requests queued up that the service gets flooded when it starts to recover.
- If the error is due to throttling and there's a window of time the service uses for throttling, continued retries could move that window out and cause the throttling to continue.
- At a certain retry threshold your app stops retrying and takes some other action, such as
  - **Custom fallback**
  - **Fail silent** (e.g. return null)
  - **Fail fast** (e.g. exception with try again later message).

## The Retry Pattern

- When the initial connection fails, the failure reason is analyzed first to determine if the fault is transient or not.
  - If the failure reason or the error code indicates that this request is unlikely to succeed even after multiple retries, then retries are not performed at all.
- Retries are performed until either the connection succeeds or a retry limit is reached.
- Implemented in many libraries such as [Entity Framework](#) and [Enterprise Library](#).

## Queues

- You can use a third-party queue to persist the requests beyond a temporary failure.
- Requests can also be audited independent of the primary application as they are stored in the queue mechanism.

### *Queue-Based Load Leveling pattern*

- Use a queue that acts as a buffer between a task and a service it invokes in order to smooth intermittent heavy loads that can cause the service to fail or the task to time out.
- Help you design solutions that handle transient errors.
- Benefits
  - **Control costs**
    - The number of service instances deployed only have to be adequate to meet average load rather than the peak load.
  - **Maximize scalability**
    - Both the number of queues and the number of services can be varied to meet demand.
  - **Maximize availability**

- Delays arising in services won't have an immediate and direct impact on the application, which can continue to post messages to the queue even when the service isn't available or isn't currently processing messages.

## Scalability Patterns

### Asynchronous Messaging

- Messaging supports asynchronous operations, enabling you to decouple a process that consumes a service from the process that implements the service.
- Asynchronous Messaging with Variable Quantities of Message Producers and Consumers
  - **Problem:** handling variable quantities of requests
  - Applications passes data through a messaging system to another service (a consumer service) that handles them asynchronously
    - Business logic in the application is not blocked while the requests are being processed.
  - In Azure: Storage Queues or Service Bus Queues
  - Most message queues support three fundamental operations:
    - A sender can post a message to the queue.
    - A receiver can retrieve a message from the queue (the message is removed from the queue).
    - A receiver can examine (or peek) the next available message in the queue (the message is not removed from the queue).

### Cached Data Consistency

- Impractical to expect that cached data will always be completely consistent with the data in the data store.
- Consider a strategy that helps to ensure that the data in the cache is up to date as far as possible, but can also detect and handle situations that arise when the data in the cache has become stale.

#### *Read/Write-Through Caching*

- If the data is not in the cache, it is transparently retrieved from the data store and added to the cache. Any modifications to data held in the cache are automatically written back to the data store as well.
  - **Cache-aside strategy:** This strategy effectively loads data into the cache on demand if it's not already available in the cache.
- Example implementation in Azure using *Redis Cache*

- i. When the web application loads for the first time it makes a request (GET) to the Redis Cache instance for the name of the featured player using the key: player:featured.
- ii. The cache will return a nil value indicating that there is not a corresponding value stored in the cache for this key.
- iii. The web application can then go to the Cosmos DB data store and gets the name of the featured player.
- iv. The name of the featured player will be stored in the Redis Cache using a request (SET) and the key player:featured.
- v. The web application returns the name of the featured player and displays it on the homepage.
- vi. Subsequent requests for the home page will use the cached version of the value as Redis Cache will successfully return a value for the name of the featured player.
- vii. If an administrator updates the featured player, the web application will replace the value in both Redis Cache and Cosmos DB to maintain consistency.

## Load Balancing

- The application traffic or load is distributed among various endpoints by using algorithms.
- Flexibility to grow or shrink the number of instances in your application without changing the expected behavior.
- 🛠️ **Load Balancing Strategy**
  - i. Decide whether you wish to use a **physical** or a **virtual** load balancer
    - 💡 Deploy virtual load balancer (hosted in VMs) if a company requires a very specific load balancer configuration.
  - ii. Select a load balancing algorithm e.g. round robin or random choice.
    - **Round-robin:** List is ordered on each request and sent to next instance.
      - No standard for deciding which address will be used by the requesting application.
      - E.g. geographically closer
  - iii. (optional) Configure state (affinity or stickiness)
    - **Stickiness** allows you determine whether a subsequent request from the same client machine should be routed to the same service instance

## Data Patterns

## Redis Cache

- Open source cache and message broker that you can deploy to support high availability.
- Key-value pair NoSQL storage.
- **!** Azure Cache: Depreciated, use Redis Cache.
- **SKUs**
  - **Basic:** Single node. Memory-sizes: 250 MB - 53 GB
  - **Standard:** Two nodes in master/replica configuration, and SLA.
  - **Premium:** On more powerful hardware, geo replication, VNet integration, and more.

## Database Partitioning

- The physical separation of data for scale.
- Any data access operation will only occur on a smaller subset (volume) of data which in turn ensures that the operation will be more efficient when compared to a query over the entire superset of data for your application.
- Partitioning also spreads your data across multiple nodes which can individually be replicated or scaled.
- E.g. **sharding for Azure SQL Database**
  - i. Manually create shards
  - ii. Use Elastic Scale feature
    - Each partition is an instance of Azure SQL Database.
    - If load to one shard gets high:
      - Uses "Split" feature to create new shards from original
    - If load to multiple shards get low
      - Uses "Merge" feature to create new single shard from multiple shards.

### *Database Partitioning Scenario*

- **Problem:** Hosting Large Volumes of Data in a Traditional Single-Instance Store
  - Limitations:
    - Finite storage space
    - Finite computing resources:
    - Finite network bandwidth
    - Geography: Single region, hard with more.
  - Scaling vertically is temporary solution that can postpone effects.
    - Cloud application should be scale almost infinitely.

- **Solution:** Partitioning Data Horizontally Across Many Nodes
  - Divide the data store into horizontal partitions or shards.
  - Each shard has the same schema, but holds its own distinct subset of the data.
  - **💡** Abstract the physical location of the data in the sharding logic
    - Provides a high level of control over which shards contain which data
    - Enables data to migrate between shards without reworking the business logic of an application.
    - The tradeoff
      - The additional data access overhead required in determining the location of each data item as it is retrieved.
      - To handle it, implement a sharding strategy with a shard key that supports the most commonly performed queries.


## 5.2. VM Availability (SLA, Availability Sets, Availability Zones)

### VM Availability


- Microsoft Azure provides a Service Level Agreement (SLA)
  - backed by a financial service credit payment for IaaS Virtual Machines.
  - Depends on the deployment of the virtual machine and what resources it uses.

### Availability Set

- Ensures SLA can be provided.
  - One VM being available at least 99.95% of the time.
- Ensures VMs you deploy within an Azure data center are isolated from each other.
- Ensures that all virtual machines that are added to the set are placed in such a way as to ensure that neither hardware faults or Azure fabric updates that is unplanned and planned maintenance events can bring down all of the virtual machines.
- Application availability can be impacted by:
  - Unplanned hardware maintenance event
  - An unexpected downtime
  - Planned maintenance events
- **💡** To reduce or remove the impact of downtime:
  - Place virtual machines in an availability set for redundancy.
  - Use managed disks for all VMs placed in an availability set.
  - Use Scheduled Events to respond to events.
  - Place each tier of your application in a separate availability set.

- Use a load balancer in combination with availability sets.
-  Avoid single instance VMs in an availability set.
  - They are subject to any SLA unless all the Operating System and Data disks are using Premium storage.

## Update and Fault Domains

- Each machine in the Availability set is placed in an Update Domain and a Fault domain.
- A **Fault Domain (FD)** is essentially a rack of servers.
  - It consumes subsystems like network, power, cooling etc.
- **Update Domain (UD)**
  - Purposeful move to take down one (or more) of your servers.
  - It will walk through your update domains one after the other.
-  FDs come in sets of 2 and UD's come in sets of 5 (default)
  - So if you deploy more than 5 VMs in an availability set they'll end up in same UD and FD.

## Multiple availability sets

- E.g. N-tier availability sets
  - An extension of the availability set model is used logically to place individual tiers of an application into separate Availability Sets.
  - E.g. put front-ends in one, and data tier in another availability set.

## Availability Zones

- Advent of a data center-wide fault would prevent the Availability set from functioning.
- Allows for a complete data center failure and keep your VM based application running.
  - Zone = separate zone or building within a single Azure region.
- You can set the count of zones while creating VM.
  - There is a maximum of three Availability Zones per supported Azure region.
  - Each Zone operates on an entirely isolated power source, cooling system, and network infrastructure.

### 5.3. Azure VM Scale Sets

#### Azure VM Scale Sets

- Allows a virtual machine to deploy up to 1000 times in the same subnet.

- Allow accurate auto-scaling
- Provides high degree of control like IaaS, but manages networking/storage/compute/load balancing like PaaS.
- Requires no pre-provisioning, automatically configures and manages:
  - Network
  - Load balancer
  - Network Address Translation (NAT)
- Handles resource creations, dependencies and configurations.

## Virtual Machines vs. Virtual Machine Scale Sets

| Functionality               | Scale set                                                  | VM                        |
|-----------------------------|------------------------------------------------------------|---------------------------|
| Azure Autocore              | 👍                                                          | 👎                         |
| Availability zones          | 👍                                                          | 👍                         |
| Reimaging                   | 👍                                                          | 👎                         |
| Overprovisioning            | 👍 Automatically increase reliability and faster deployment | 👎 Custom code is required |
| Upgrade policy              | 👍 Can upgrade all VMs in scale                             | 👎 Must be orchestrated    |
| Attach data disks           | 👍 Applies to all instances in data sets                    | 👍                         |
| Attach non-empty data disks | 👎                                                          | 👍                         |
| Snapshot                    | 👎                                                          | 👍                         |
| Capture image               | 👎                                                          | 👍                         |

| Functionality                | Scale set                | VM                |
|------------------------------|--------------------------|-------------------|
| Migrate to use managed disks | 👉                        | 👉                 |
| Assign public IP addresses   | 👉 Requires load balancer | 👉 Possible on NIC |

## Connecting to a VM Scale Set instance VM

- Done by accessing Load balancer inbound NAT rules and using the correct IP address and custom port.
- You can see & set it in Load Balancer -> Inbound NAT Rules.

## Continuous Delivery in VMSS

- By default, the pipeline builds code, and updates VM scale set with the latest version of your application.
- Can be done by two ways:
  - Immutable Deployment**
    - Create a custom image that already contains the OS and application in a single VHD.
    - **Advantages**
      - Predictability
        - Any new versions of the application can be tested on a similar VM Scale set and then deployed directly into the production instances without any downtime.
      - Easy to scale
      - Easy to roll-back
      - Faster to scale (no code to install on each VM as it is deployed)
      - Can use toolset from Visual Studio Team Services
  - Use of VM extensions to install software to each instance at deployment time.
    - Custom script VM extension to install/update your application on VM scale set

## Large VM Scale Sets



- ! Scale sets created from Azure Marketplace images can scale up to 1,000 VMs.
- ! Scale sets created from custom images can scale up to 300 VMs.
- Layer-7 load balancing with the Azure Application Gateway is supported for all scale sets.
- Scale sets are defined with a single subnet:
  - ! Ensure subnet is large enough to handle all potential VM instances.

## Large scale set: can scale beyond 1000 VMs

- Requires `singlePlacementGroup = false` property setting.
  - Layer-4 load balancing with scale sets composed of multiple placement groups requires Azure Load Balancer Standard SKU.
  - **Fault Domains** and **Update Domains** relate to a single placement group, to maintain high availability ensure there are at least two VM instances in each Fault Domain and Update Domain.
- ! Large scale sets require Azure Managed Disks.
- Ensure your compute limits are high enough, the requirement for compute cores will prevent a successful deployment if not.

## 5.4. Hybrid Cloud

### Hybrid Cloud

## Azure AD

- In its basic form, it's a free service that provides the ability for Single Sign-On into cloud applications.
- Allows connection with
  - consumers (**Azure AD B2C**)
  - business partners (**Azure AD B2B**) without deploying complex infrastructure or federation.

### Azure AD Connect

- Provides the ability to integrate on-premises directories with Azure AD.
- Provide single sign-on to cloud applications such as Office 365, Azure and other SaaS applications.

## Azure AD Hybrid solutions

- Azure AD Connect provides the choice of:
  - **1. Password Synchronization only** , the ability to synchronize users and groups.
  - **2. ADFS** , to allow on-premises authentication, 3rd party MFA, etc.
  - **3. Pass-through authentication** , provides on-premises authentication without deploying ADFS.
- In all 3, you need Azure AD Connect to provide the synchronization engine.
  - Additionally, the **Microsoft Identity Manager (MIM)** can be installed on-premises and used to **configure** the users, groups and attributes to be synchronized.

## Azure AD Pricing

- **Basic:** SLA + self-service password reset for cloud users.
- **Premium:** MFA + other stuff.
- **P2 only:** Conditional risk policy, privileged identity management

## Azure AD Domain Services

- Provides managed domain services.
- Windows Server Active Directory compatible.
  - LDAP, Kerberos, NTLM, Group Policy, and domain join capabilities are compatible.
  - **!** Not all features available in Windows Server AD are available in Azure AD Domain Services.
- Compatible with both Cloud only tenants and hybrid tenants using Azure AD Connect.
- No additional costs other than underlying Azure IaaS Virtual Machines.

## Cloud Only Tenant

- Managed by Azure.
- All user identities, credentials and groups, including group memberships, are created and managed in Azure AD.
- All the Azure AD objects are available within this domain

## Hybrid Cloud Tenant

- Synchronized to an on-premises directory

- An additional stand-alone Domain is created by the managed service.
  - The managed domain is a standalone domain and not an extension to the on-premises directory.
- Management
  - Tenant identities are still created and managed within Azure AD
  - On-premises identities are still created and managed on-premises.
- All objects from the on-premises domain and the Azure AD tenant are available to the managed service domain.
- Allows users to sign in to cloud services with their on-premises identities
  - Azure AD Connect must be configured to allow password synchronization
    - Allows resources in the cloud to connect to the managed domain can use Kerberos to authenticate.
- Managed by Azure
  - So domain administrator does not need to
    - manage this domain or any domain controllers.
    - have has no domain or enterprise admin privileges.
  - Available automatically, no need for AD replication.

## Azure AD Pass-Through Authentication

- Enables users to sign in to both on-premises and cloud applications using the same credentials.
- Azure AD validates users' passwords against on-premises Active Directory.
  - passwords are never stored in the cloud.
- No need to deploy ADFS infrastructure.
  - No need for complicated certificates or trusts.
- 🛠️ Azure AD Connect installs **Pass-through authentication agent** on same server where it is.
  - 💡 Install additional agents to make the service highly available.
- Free for all Azure AD tiers.
- Multi-forest environments are supported with some required routing changes.
- Users can be enabled to use self-service password management from the Azure AD directly.
- No additional ports or network configuration is required since the agent communicates outbound, so no perimeter network is required.
- Takes advantage of Azure AD Conditional Access policies, including Multi-Factor Authentication (MFA).

## 5.5. Networking Azure Application Components

### Networking Azure Application Components

#### Virtual Networks

- VNet: the logical unit of multiple or all network resources in an Azure region.
- Within a VNet, you create one or more **subnets**.
  - **!** All traffic within a subnet is allowed, but communication across different subnets is blocked by default
- Several Azure Resources are making use of VNets and subnets for IP addressing
  - IaaS e.g. Virtual Machines, VM Scale Sets, Load Balancers, Azure Traffic Manager,...
  - PaaS e.g. Service Fabric, Azure Container Services, Hadoop, Azure Application Services,...

#### Network Security Groups (NSGs)

- Software-defined firewalling
- Protects incoming traffic to Azure public IP addresses.
- Top level object that is associated to your subscription.
- Rules
  - **Inbound rules** are applied on the incoming packets to a VM.
  - **Outbound rules** are applied to the outgoing packets from the VM.
  - Can be changed at any time, and changes are applied to all associated instances.
  - An incoming or outgoing packet has to match an Allow rule for it be permitted, if not it will be dropped.
  - By default
    - Connectivity to the internet is allowed for Outbound direction.
    - Blocked for Inbound direction"
  - **!** The default rules cannot be deleted, but because they are assigned the lowest priority, they can be overridden by the rules that you create.
    - For example, a rule with a lower priority number (e.g. 100) is processed before rules with a higher priority numbers (e.g. 200). Once a match is found, no more rules are processed.
- Can be assigned to both VNet and VM (individual NIC).
- **!** Limitations:
  - A VM or subnet can be associated with only 1 NSG, and each NSG can contain up to 200 rules.

- You can have 100 NSGs per subscription.
- NSGs cannot span Azure Regions.

## Multi-Region Virtual Network Architecture

- Applies for cross regions and on-premises
- 💡 From the Public Internet, Azure allows for load balancing across multiple Azure Regions by deploying Azure Traffic Manager.
- **Interconnecting Azure Regions** with each other is possible in three different ways:
  - i. Configuring Azure Site-to-Site VPN between both regions
    - Encrypted tunnel connection over internet.
  - ii. Configuring Azure ExpressRoute communication tunnels
  - iii. Azure VNET Peering
    - Networking without encryption.
- **Interconnecting Azure Regions** with on-premises data centers is possible in two different ways:
  - i. Configuring Azure Site-to-Site VPN between Azure and on-premises
  - ii. Configuring Azure ExpressRoute communication tunnels between Azure and on-premises

## IP Addressing

- IP-address gets allocated to a NIC during provisioning of the NIC
- First available IP-address in a subnet range is x.x.x.4
- Azure Subnets support dynamic (=default) and static IP addressing
- **Public IP-addressing**
  - Used for all public internet-facing communication
  - Required parameter when creating a VM from the portal
- **Private IP-addressing**
  - Used for all inter-VNET communication
  - Used for all communication between an Azure VNET and an on-premises VNET

## Azure DNS Resolving

- DNS server settings are configured on VNET level
- Using Azure DNS is the default configuration setting, but this can be modified
  - Use your custom DNS configuration:
    - Azure DNS Appliance (from Azure MarketPlace)

- Azure VM (e.g. Windows ADDS with DNS)
  - On-premises DNS solution (requires connectivity)
- Public DNS names (available for VMs and App Services) must be unique across Azure regions.
  - An example of such Public DNS name is `host.region.cloudapp.azure.com`

## Load Balancing

### Azure Load Balancer

- Uses private IP addresses in back-end pools.
  - You can associate it to a Availability set, Single virtual machine or Virtual machine scale set
    - 💡 In an availability set, it allows it to grow to 10x the number of instances (100 in Basic to 1000 in Standard in a single Availability Set).
- Can handle almost any TCP or UDP traffic e.g. RDS (Remote Desktop Services Farm), Linux SSH.

#### *Azure Load Balancer Types*

- Can be configured both as an **external load balancer** or as an **internal load balancer**
  - where one cannot act as both external and internal at the same time.

#### External load balancer

- The load balancer front-end is configured with a Public-facing IP-address
- Sends all traffic along to the back-end pool servers, using their internal IP-addresses.

#### Internal Load Balancer

- Doesn't have a Public-facing IP-address, and all communication is based on internal IP-addressing and IP-routing.
- Used for e.g. database server backends

*Azure Load Balancer SKUs*

| Basic                                      | Standard                                                                         |
|--------------------------------------------|----------------------------------------------------------------------------------|
| Up to 100 backend instances                | Up to 1000 backend instances                                                     |
| Single Availability Set                    | Availability Sets are not required; providing support for Availability Zones     |
| Basic NAT and Probe health status          | Integrated Frontend and Backend health metrics                                   |
| No HA Ports                                | Support for HA Ports                                                             |
| Network Security Groups (NSG) are optional | Network Security Groups are required during configuration and deployment         |
| Free                                       | Charged hourly based on number of rules configured (except NAT) & data processed |

## Azure Application Gateway

- Load Balancing, active on Layer 7 of the network stack; this mainly means it is "application intelligent."
- Main features Application Gateway provides, compared to Azure Load Balancer, are:
  - HTTP/HTTPS traffic only, no other ports allowed
  - SSL Offloading
  - Cookie Affinity
  - Web Application Firewall (WAF)
  - URL Based Routing
- SSL Termination
  - **SSL Offloading**, by importing the SSL Certificate onto the App Gateway; traffic to the backend servers don't require HTTPS communication, also that would still be an option.
  - **HTTP to HTTPS redirect** ; this means that, whenever a user is connecting to the web app using HTTP, the request will be redirected to HTTPS, forcing SSL Tunneling for this given request.

- **Web Application Firewall (WAF)**
  - Protection against several common attacks and threats on application workloads.
    - E.g. SQL Injection, Cross-site scripting, Protocol violations, Generic attacks, HTTP rate limiting, Scanner detection, Session fixation, LFI/RFI.

## Azure Marketplace Load Balancing Appliance

- Recommended for more capabilities around management, monitoring and control.
- Support is initially provided by Microsoft, backed by SLAs, and acting as a SPOC (Single point of contact).

### *Licensing Models*

#### Bring Your Own License (BYOL)

- This is an ideal candidate if you are removing or downsizing on your on-premises running third-party load balancer.
- Depending on the specific licensing terms of the vendor, one can reuse the license key on the Azure VM Appliance.

#### Pay-Per-Use

- The monthly Azure VM consumption cost is based on the VM Size allocation to the Appliance, as well as a monthly licensing fee for the third party load balancing application within the VM.

## Azure Traffic Manager

- Control the distribution of user traffic to your specified endpoints.
- Works by applying an intelligent policy engine to Domain Name System (DNS) queries for the domain names of your internet resources.
- Flexible as it allows you to mix various endpoints behind the same DNS name.
- Easy to integrate with Web Apps.

## Example Azure Traffic Manager use cases

- **Failover**
  - Can poll to determine if and endpoint is online or offline.
  - Traffic is routed in the next online endpoint that's highest in the priority list.
- **Geography** Uses Internet Latency Table to find closes endpoint to the client.



- **Distribution**
  - Near-random way to distribute evenly across set of endpoints.
  - Distribution can optionally be weighted.
    - ⚠ Weighting is good if you have a smaller recovery site and want to keep majority of the traffic to the primary site using larger service tiers.

## Azure Traffic Manager Workflow

1. User traffic to company domain name, e.g. fabrikam.com
2. Company domain name to Traffic Manager domain name
  - CNAME resource record that maps the company domain name (e.g. fabrikam.com) to traffic manager domain name (e.g. fabrikamweb.trafficmanager.net)
3. Traffic Manager domain name and profile: The user's DNS server sends a new DNS query for the Traffic Manager domain name (e.g., fabrikamweb.trafficmanager.net), which is received by the Traffic Manager DNS name servers.
4. Traffic Manager profile rules processed: load balancing method and monitoring status to determine endpoint.
5. Endpoint domain name to user:
  - Returns CNAME that maps Traffic Manager domain name to the domain name of the endpoint.
  - Users DNS server resolves domain name to IP and sends it to user.
6. User calls the returned endpoint directly by using its IP address.

## Connectivity options

### On-Premises to Azure Connectivity

- This can be between an on-premises network and one or more Azure regions, or between multiple Azure regions

| Connectivity | Benefits                                                                                                                                                                                                                                                               |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ExpressRoute | <ul style="list-style-type: none"> <li>• ⚠ Use ExpressRoute as primary cross-premises connectivity.</li> <li>• Multiple circuits for redundancy &amp; better routing.</li> <li>• ⚠ Use ExpressRoute-VPN co-existence for highly available, redundant paths.</li> </ul> |

**Connectivity****Benefits**

Site-to-Site  
VPN

- 💡 S2S VPN over internet for remote branch locations
- BGP & active-active configuration for HA and transit

Point-to-Site  
VPN

- P2S VPN for mobile users & developers to connect from anywhere with macOS & Windows
- AD/radius authentication for enterprise grade security

- **Multi-Region VPN Connectivity**

- Forced Tunneling

- Azure traffic can be rerouted to an on-premises virtual network, to be routed through an existing Site-to-Site VPN or ExpressRoute, into the internal Azure VNET.
    - Security improvement as internal Azure VMs are no longer accessible through the public internet.

## Securing Access to PaaS Services

- Goal is to disable access to PaaS services from public internet.

### *Service Endpoints*

- Use VNET **service endpoints**, where you define which PaaS services are no longer accessible through the public internet
- **Virtual network service endpoint**
  - ! Only available to Azure Storage Accounts, SQL DB Services in PaaS and Web Apps.
  - Provides the identity of your virtual network to the Azure service.
  - Service endpoints are configured on a subnet in a virtual network
  - You can configure multiple service endpoints for all supported Azure services on a subnet.
  - VNets and resources can be in different subscriptions and regions with exceptions:
    - ! SQL VNet must be in same region.
    - ! Storages Storage Account must be in same region.
- Flow:
  - i. Connect storage account to a VNet.
  - ii. Add service endpoints to the VNet.

- iii. Once service endpoints are enabled in VNet, you can secure Azure service resources to your virtual network by adding a VNet rule to the resources.
- **Access from on-premises**
  - You must also allow public (typically NAT) IP addresses from your on-premises or ExpressRoute.
  - Those IP addresses can be added through the IP firewall configuration for Azure service resources.

## VNET Peering

- Uses the Microsoft Backbone (not the public internet).
- Communication relies on internal IP addressing.
- Primary features
  - Allows you to interconnect 2 Azure VNET as if they are 1 large VNET.
  - Possible within the same Azure region, or across Azure regions
  - Supported to interconnect an Azure Classic VNET with an ARM VNET (e.g., for migrating workloads).
- 💡 If VNET peering is not an option, because you might want to encrypt your traffic within the VNET tunnel, deploy a VPN Gateway on both Azure Regional VNETs to create a Site-to-Site VPN tunnel across those regions.

## 5.6. Messaging Services

### Messaging Services

- Includes Azure Storage Queues, Service Bus Queues, Service Bus Relay, IoT Hubs, Event Hubs, and Notification Hubs
- Integration services include Azure Functions and Logic Apps.

## Event Messaging


### Storage Queues

#### *Storage Components*

- **Storage account**
- **URL:** E.g. `http://[account].queue.core.windows.net/queue`

- **Queue:** A queue contains a all of messages.
- **Message:** A message, in any format, of up to 64KB.

### *Storage Queue Message Handling*

-  Processors/consumers must be designed to not have side effects from processing the same message multiple times.
  - E.g. if it inserts to SQL it should check first if record exists.
  - Service Bus Queues on the other hand guarantees that a message is handled at least and most by single customer.
- Operations:
  - **Create/Delete Queue:** Client SDKs, PowerShell or the REST API
  - **Measure Queue Length**
    - You can get an estimate of the number of messages in the queue
    - **!** This count is not guaranteed and should be treated in your application as an approximate queue length.
  - **Insert Message into Queue:** Either a string (in UTF-8 format) or a byte array.
  - **Retrieve the Next Message**
    - A copy is retrieved
    - Message is made invisible for a specific duration
      - After duration the message is available to other queue consumers.
  - **Extend Message Lease:** You can return to the queue and update the invisibility duration for the queue message
  - **Peek at the Next Message:** Does not make message invisible
  - **Update a Message**
    - The contents of a retrieved message can be updated in the queue
    - Used e.g. with checkpoints or state for queue messages.
  - **Delete a Message:** Otherwise invisible will timeout and message will be reprocessed.

## Service Bus

- Infrastructure for **communication, event distribution, naming and service publishing**.
- Provides connectivity options for *Windows Communication Foundation (WCF)* and other service endpoints e.g. REST
  - Endpoints can be located behind network address translation (NAT) boundaries, or bound to frequently-changing, dynamically-assigned IP addresses, or both.
- Partitioned into namespaces as each namespace provide both a service and security boundary.
- Uses pull-model

- ! Can be integrated with Event grid to have push model as event grid uses push model
  - E.g. if a function listens a Service Bus then it needs to run continuously all the time, but using Event grid would allow utilizing Service Bus but only triggering function once event is received.
  - Read more: [Azure Service Bus to Event Grid integration overview | Microsoft docs](#)

### *Messaging capabilities*

#### Relayed messaging pattern (relays)

- Direct one-way messaging
- Request/response messaging
- Peer-to-peer messaging

#### Brokered messaging pattern

- E.g. topics (one-to-many), queues (one-to-many), Event Hub.
- Available through APIs and SDKs over HTTP.
- Asynchronous messaging with Queues, Topics and Subscriptions
- **Publish-subscribe and temporal decoupling**
  - Senders and receivers do not have to be online at the same time.
    - Messages are stored until they are received.

#### *Service Bus Queues*

- Extends Storage queues.
- Implements FIFO.
- Guarantees that message is received and processed both at least and at most once by the message consumers
- Flow: Message sender (*e.g. web app, mobile app and service*) => Namespace (*contains queue*) => Message receiver (*service or application*)

#### *Service Bus Relay*

- Connects existing services to new client applications without exposing the true location or address of the service.
- Supports direct peer-to-peer communication
- Advantages / Use-cases:

- On-prem resources can be revealed without them having a public IP address but using outbound connection.
  - Abstraction helps with having same URI pointing out different resources.
- Flow: Application <=> Service Bus Relay <=> Client1, Client2

## Event Grid

- Single service designed to manage and route systemic events from any source service in your Azure subscription.
- Uses push model
  - Removes need to polling
    - Triggers applications when needed
    - Pay per event: Consume compute only when necessary.
  - Can be integrated with e.g. Service Hub to pull the events and push it to e.g. functions
- Can use custom workloads or pre-determined events with each Azure service.
- Reliability through the use of a 24-hour retry with exponential back-off.
- Example ways to use
  - **Integration:** Integrate various components of your workloads together using a publish-subscribe mode.
  - **B2B:** Enable your solution to listen to events from third-party B2B services of publish events for the third-party services to consume.
  - **Compute:** Create serverless compute that is triggered by a specific Azure service event such as the creation of a database or VM.
  - **Ops automation work:** Automate the deployment of resources by subscribing to ARM events.
    - E.g. a user can request Event Grid to notify Azure Automation when a virtual machine starts or a SQL Database is spun up.
      - Events can be then used to e.g.
        - Tag virtual machines
        - File work items
        - Put metadata into operations tools
        - Automatically check to ensure services configurations are correct.
  - **Router:** You can use filters to route specific events to different or even multiple endpoints.
    - You can route to a webhook endpoint or event handler.

## Integration

## Serverless Integration

- Serverless applications: Azure Logic Apps, Azure Functions.
- Development: IDE support, visual debug history, local development, verbose debugging.
- Platform:
  - **Functions:** Developer Tooling, Bindings and triggers, open source
  - **Logic Apps:** Visual designer, 200+ connectors, functions orchestration
  - Even more such as data/storage, messaging, gateway connectors, intelligence, bots

### *Notification Hubs*

- Service infrastructure and a set of client libraries that allows you to publish push notifications.
- Can send to specific user, all users or segmented users.
- Abstracts the implementation of the various Platform Notification Systems (PNS) for each mobile platform with a single method call.
  - **Devices** are only responsible for registering PNS handles.
  - **Backend** is responsible for sending platform-independent messages to users or interest groups.

### Notification Hubs Advantages

- Multiple platforms
- Works with any backend
- Scale
- Rich set of delivery patterns
- Personalization: Each device can have one or more templates to achieve per-device localization and personalization without effecting the back-end code

### Platform notification system flow

1. Client application contacts the PNS to retrieve its handle.
2. Application stores the handle for later usage.
3. App back-end contacts the PNS by using handle to target a client application.
4. PNS forwards notification to the device specified by the handle.

### Notification hub flow

1. Client reaches out to the PNS through Notification Hubs SDK and registers PNS handle.
2. Alternatively, client sends PNS handle to application backend to have the application register the service.
3. Application back-end sends message to Notification hubs service, service forwards it to appropriate target client by using their PNS handles.

## Internet of Things (IoT)

### Event Hubs

- Event ingestion service
- Uses pull-based model
  - Can be integrated with Event Grid to allow a push model
- Can be used as a "front line" for an event pipeline of a solution architecture, sometimes known as an **event investor**
  - **Event investor**
    - a service or component
    - sits between the event consumers and publishers
    - separates the production of an event stream that obtained said events.
- Capable of publishing via AMQP 1.0 or HTTPS
- It can capture Event Hubs streaming data to store in an Azure Blob storage account.
  - Allows e.g. :
    - to each consumer to read a specific subset of the event stream
    - to consumer to act independently
- Uses **SAS (Shared Access Signature)** tokens
  - Can identify and authenticate the event publisher through SAS tokens.
- **Flow:** Event Producers --HTTP,AMQP--> Partitions -> Consumer Groups -> Event Recievers.

### Event hubs pricing

- Throughput units or Pre-purchased units of capacity
- Throughput units can include the capacity of either 1 MB per second or 100 events per second
- Up to to 2 MB per second if its egress
- Units are billed hourly for a minimum of one hour, and up to a maximum of 20 throughput units per Event Hubs namespace.



## Event Hubs scenarios

- Process streams
- Save in long/term storage e.g. *service bus, Azure DBs, HDInsight, Azure Storage, Azure Data Lake*
- Present take actions through *Power BI Dashboards, Search and query, Cortana analytics, Devices to take action*

## IoT Hubs

- Provides reliable and secure bi-directional communication between a multitude of IoT devices with a solution back end.
- **Message routing** option built in that sends the message to other Azure services
- More advanced & device specific than Event Hubs
  - Can provide additional features such as Device twins, which can be used to store and research device state information.
  - Event Hubs handles only event ingress communication (device-to-cloud) while IoT provides device-to-cloud and cloud-to-device communication.
- You can use an IoT Hub to receive messages from a device and then push them to a Logic App for processing
  - ! Up to 10 endpoints are supported.

### *IoT Hubs connection and security*

- Uses outbound connection only from devices.
- The path between device and service must be secured at application protocol layer.
- Requires gateway that can be either:
  - **Protocol gateway**
    - Always deployed in the cloud
    - With a protocol gateway protocol, translations are carried out, such as MQTT to AMQP.
  - **Field gateway**
    - Deployed locally with a device.
    - You can make time-sensitive decisions, run analytics on edge, provide device management services or even enforce security and privacy constraints.

## Event Hubs vs IoT Hubs

| Area                            | IoT Hub                                                                                                                                                                                                                                                                                                        | Event Hubs                                                                                                                                                                                                                                   |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Communication patterns</b>   | <ul style="list-style-type: none"> <li>• Device-to-cloud communications (<i>messaging, file uploads, and reported properties</i>)_</li> <li>• Cloud-to-device communications (<i>direct methods, desired properties, messaging</i>).</li> </ul>                                                                | Only enables event ingress (usually considered for <i>device-to-cloud scenarios</i> ).                                                                                                                                                       |
| <b>Device state information</b> | Device twins can store and query device state information.                                                                                                                                                                                                                                                     | No device state information can be stored.                                                                                                                                                                                                   |
| <b>Device state protocol</b>    | <ul style="list-style-type: none"> <li>• Supports MQTT, MQTT over WebSockets, AMQP, AMQP over WebSockets, and HTTPS.</li> <li>• Works with the Azure IoT protocol gateway, a customizable protocol gateway implementation to support custom protocols, authentication, message transformations etc.</li> </ul> | Supports AMQP, AMQP over WebSockets, and HTTPS.                                                                                                                                                                                              |
| <b>Security</b>                 | Provides per-device identity and revocable access control. See the Security section of the IoT Hub developer guide.                                                                                                                                                                                            | Provides Event Hubs-wide shared access policies, with limited revocation support through publisher's policies. IoT solutions are often required to implement a custom solution to support per-device credentials and anti-spoofing measures. |
| <b>Operations monitoring</b>    | Enables IoT solutions to subscribe to a rich set of device identity management and connectivity events such as individual                                                                                                                                                                                      | Exposes only aggregate metrics.                                                                                                                                                                                                              |

| Area                                        | IoT Hub                                                                                                                                                                                      | Event Hubs                                                                                                                                       |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
|                                             | device authentication errors, throttling, and bad format exceptions. These events enable you to quickly identify connectivity problems at the individual device level.                       |                                                                                                                                                  |
| <b>Scale</b>                                | Is optimized to support millions of simultaneously connected devices.                                                                                                                        | Meters the connections as per Azure Event Hubs quotas. On the other hand, Event Hubs enables you to specify the partition for each message sent. |
| <b>Device SDKs</b>                          | Provides device SDKs for a large variety of platforms and languages, in addition to direct MQTT, AMQP, and HTTPS APIs.                                                                       | Is supported on .NET, Java, and C, in addition to AMQP and HTTPS send interfaces.                                                                |
| <b>File upload</b>                          | Enables IoT solutions to upload files from devices to the cloud. Includes a file notification endpoint for workflow integration and an operations monitoring category for debugging support. | Not supported.                                                                                                                                   |
| <b>Route messages to multiple endpoints</b> | Rules determine how messages are routed to custom endpoints. ! Up to 10 custom endpoints are supported.                                                                                      | Requires additional code to be written and hosted for message dispatching.                                                                       |

## Time Series Insights

- Built for visualizing, storing, and querying vast amounts of time series information, such as information generated by IoT devices.
- Requires no upfront data preparation.

- It can obtain and store millions of sensor events per day with a one-minute latency, giving it the ability to provide near real-time insights.

#### *Time series data*

- Shows how an asset or process changes over time.
  - It has a timestamp that is unique to its kind which is most meaningful as an axis.
- Arrives in time order and can usually be an insert rather than an update to your database.
- Because it obtains and stores every new event as a row, the changes can be measured over a time frame, enabling one to not only look back into the past but also predict the future with the data.

#### *Use cases for Time Series Insights*

- Storing and maintaining *time series data* in a scalable format.
- Near real-time data visualization
  - When you connect an event source, the event data can be viewed, queried, and explored.
- Producing customer application with REST API.
- Global view of time series data streaming from different locations for multiple sites or assets
  - Allows you to connect multiple event sources to the environment

*Disclaimer: All data and information provided on this site is for informational purposes only. This site makes no representations as to accuracy, completeness, correctness, suitability, or validity of any information on this site & will not be liable for any errors, omissions, or delays in this information or any losses, injuries, or damages arising from its display or use. All information is provided on an as-is basis.*