
Linux Academy
The Blueshift Guide
AZ-300: Microsoft Azure Architect Technologies
Enter

Subscription and Services Layer

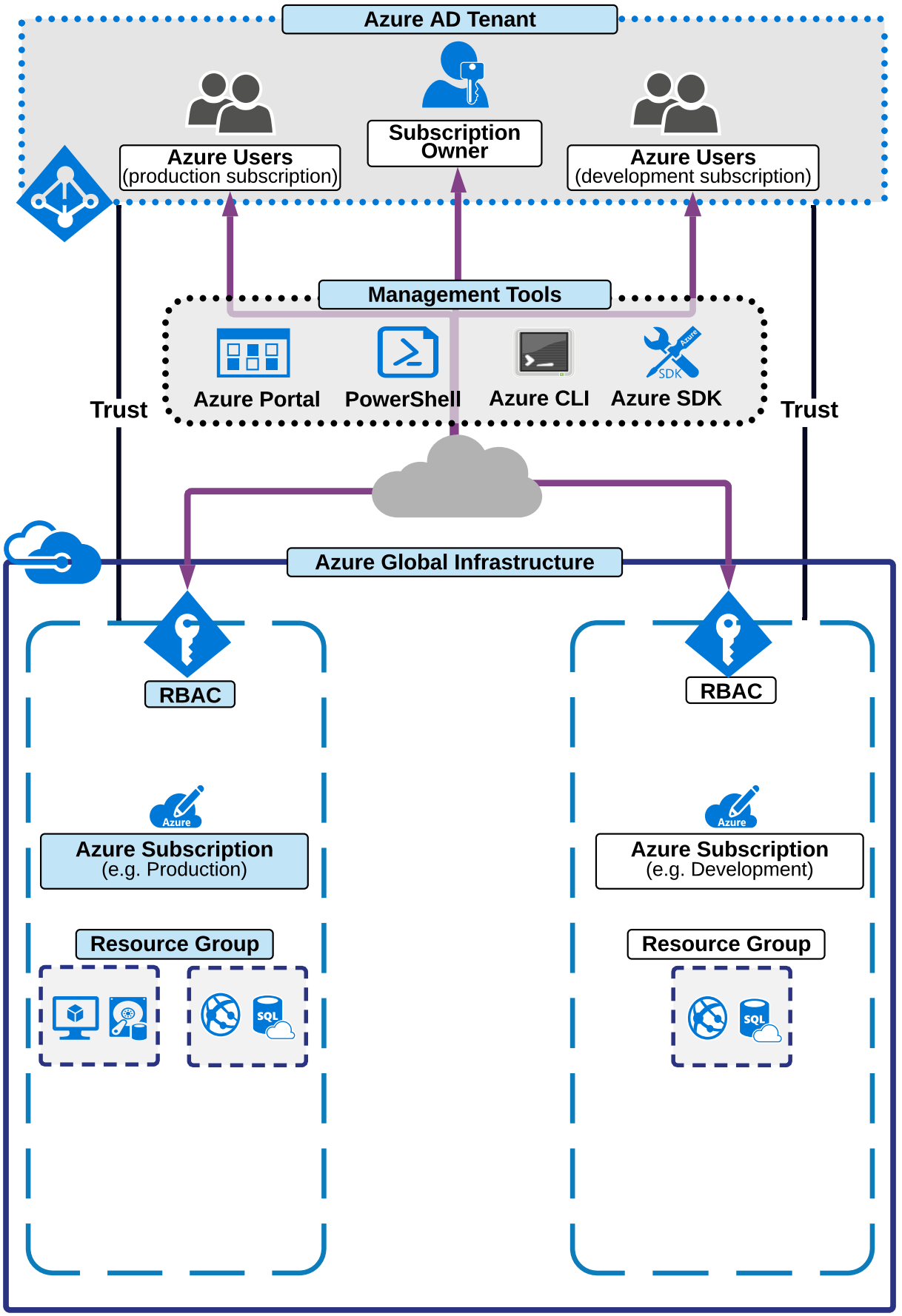
Physical and Networking Layer

## Subscription and Services

Within Azure, the subscription is where all resources are ultimately contained. This is the uppermost billing and management layer.

This page provides a high-level view and recap of key components and tools that will be used throughout this course.

Appendix



Subscription and Services Layer

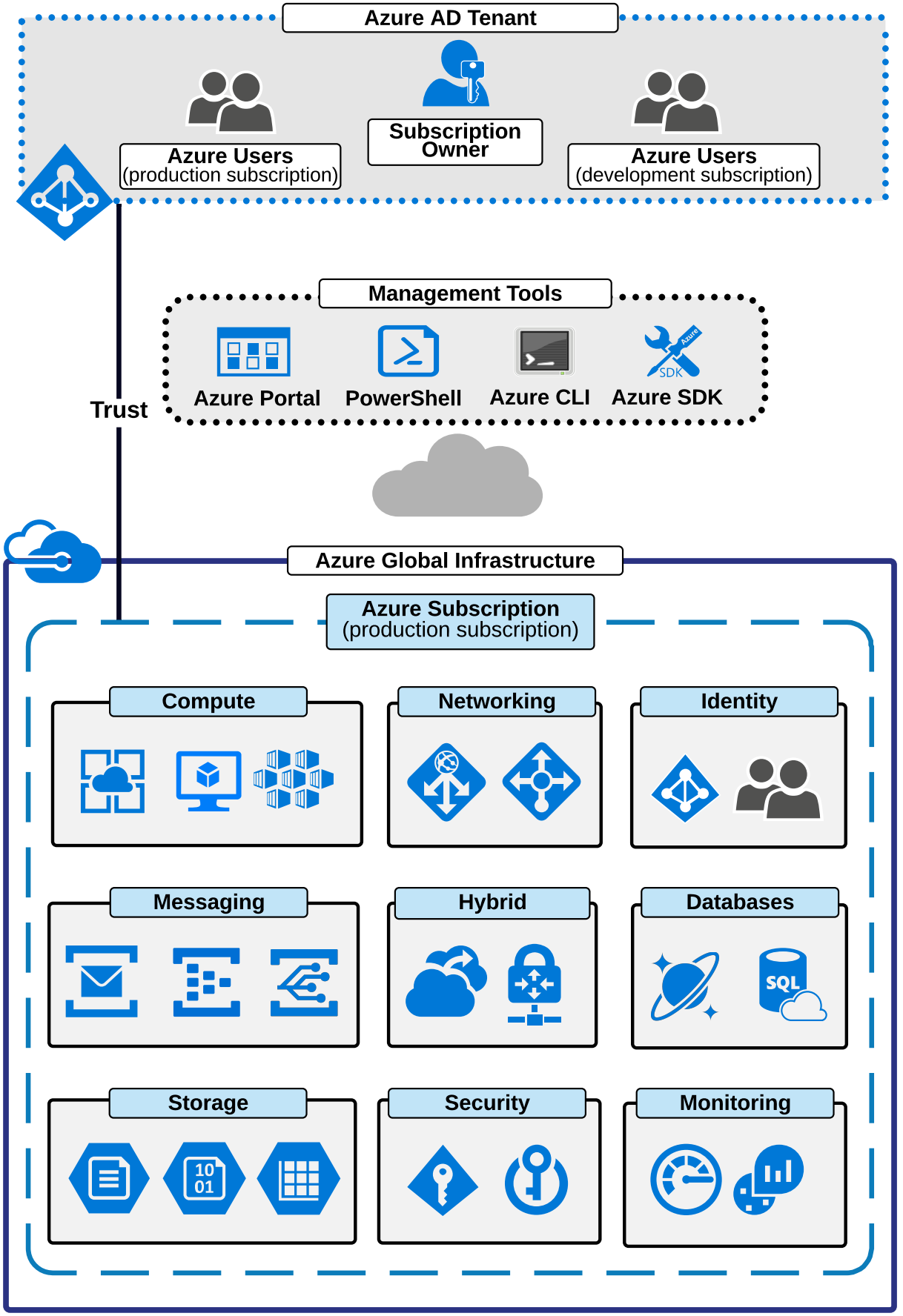
Physical and Networking Layer

## Account and Services
(Subscription)

Within the Azure subscription, represented here as "production subscription," you can see the services you must be familiar with for the AZ-300 exam.

Go Back

Appendix



## Azure Subscription

Close

Azure subscriptions are the absolute root-level container of any resource within Azure. You must first have a subscription before any Azure services can be created.

**Here are some important reasons to use an Azure subscription:**

- As a way to organize your resources at the billing and management layer
  - For example, one subscription for production purposes and a separate one for development
  - Another example may be to have one subscription for each internationally separate business unit

Resources, costs, and metrics can be viewed for all resources across your subscription.

Appendix

## Resource Group

Close

For any resource you wish to create within Azure, it must be based within a resource group. Resource groups can act as a logical point for managing security and grouping resources with similar lifecycles.

**Here are some important notes about resource groups:**

- A resource group can contain resources that are based in different regions.
- A resource group can be used to apply security policies, controlling administrative access.
- The location of a resource group does not limit the location of resources within that resource group — instead, it specifies the location of the resource group metadata.

## Azure AD Tenant

Close

The Azure AD tenant, based on Azure Active Directory, is where identity and access is managed for Azure as well as a range of other Microsoft Cloud services. An Azure AD tenant refers to a single directory (for example, linuxacademy.onmicrosoft.com).

**Here are some important notes about the Azure AD tenant:**

- An Azure AD tenant can be associated with multiple subscriptions — this is useful when your company wishes to use separate subscriptions for production and development, while maintaining a single identity.
- Azure subscriptions can only be associated with one AD tenant.

## Azure Global Infrastructure

Close

Azure global infrastructure is the underlying physical infrastructure that powers Azure. This is managed by Microsoft and includes data centers and network connectivity across the world.

More information can be found on the key components that make up the Azure global infrastructure within the Physical page found here.

## Management Tools

Close

Microsoft provides a range of tools for creating, accessing, managing, configuring, and monitoring resources within Azure. These are available across multiple platforms, including Windows, Linux, and macOS.

Below is a summary of some of the important tools:

- Azure Portal: Comprehensive management of Azure services through a web browser (portal.azure.com)
- PowerShell: Various cmdlets available to create, test, deploy, and manage Azure solutions
- Azure CLI: Lightweight cross-platform command line tool with functionality similar to PowerShell
- Azure SDK: Tools and templates to help develop Azure solutions across various programming languages

All tools ultimately interact with Azure through the Azure Resource Manager API.

Subscription and Services Layer

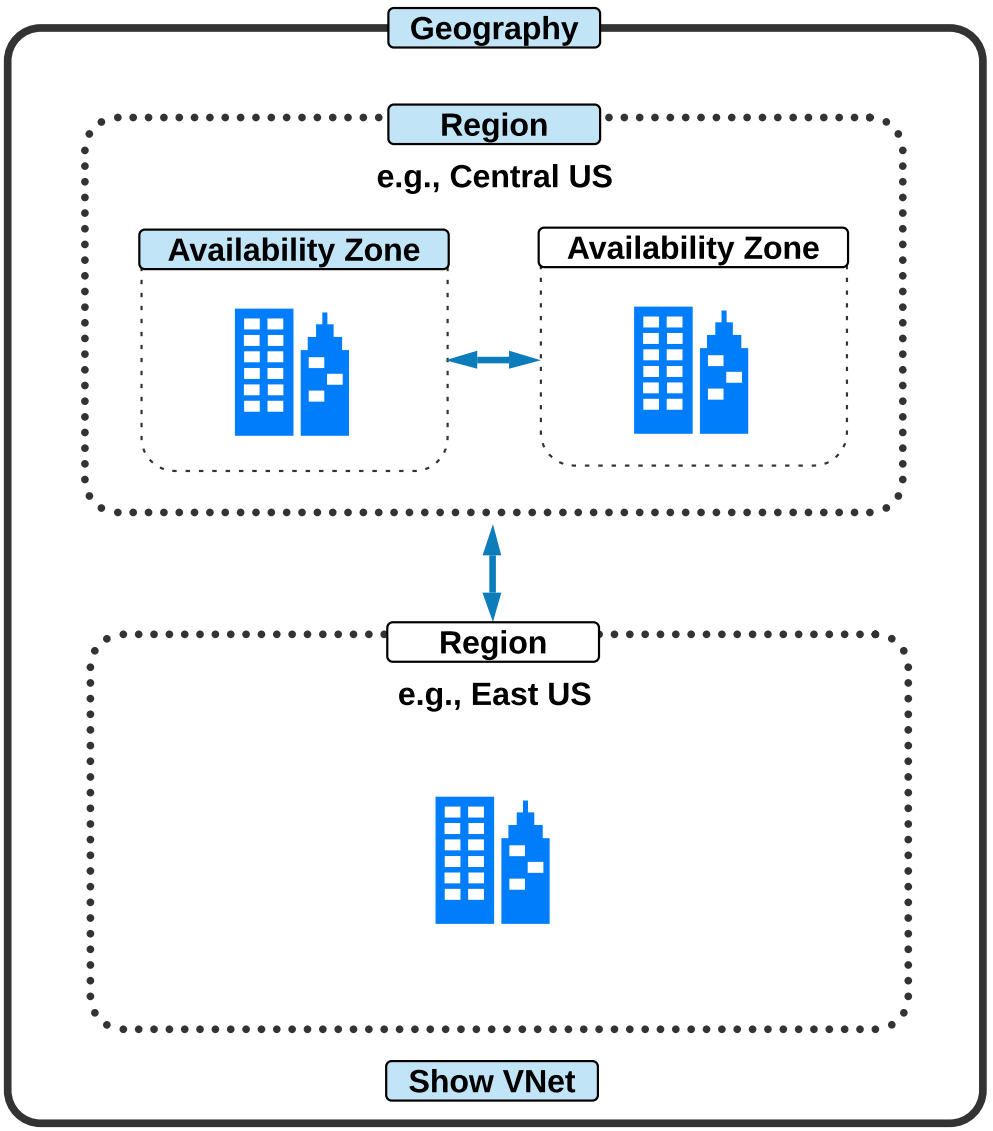
Physical and Networking Layer

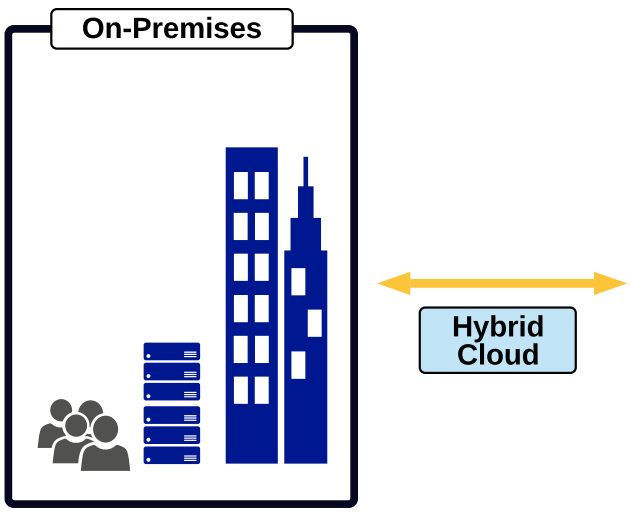
## Physical and Networking

The Physical and Networking section provides an overview of the Microsoft global infrastructure, which is used to back all Azure services.

This page provides an overview of the main components and categories of the Azure infrastructure.

Appendix



Subscription and Services Layer

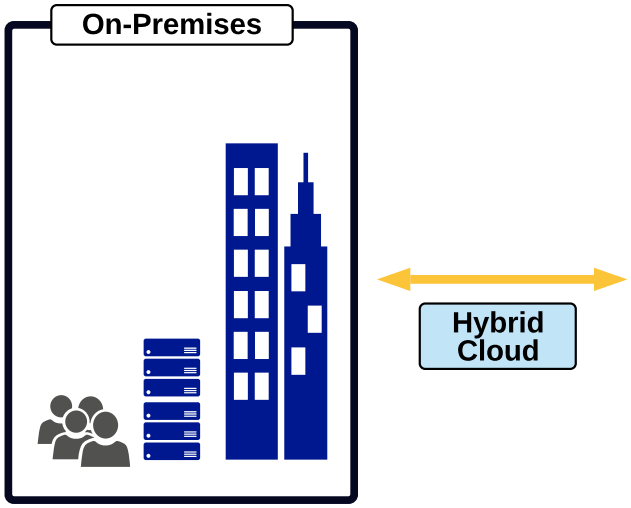
Physical and Networking Layer

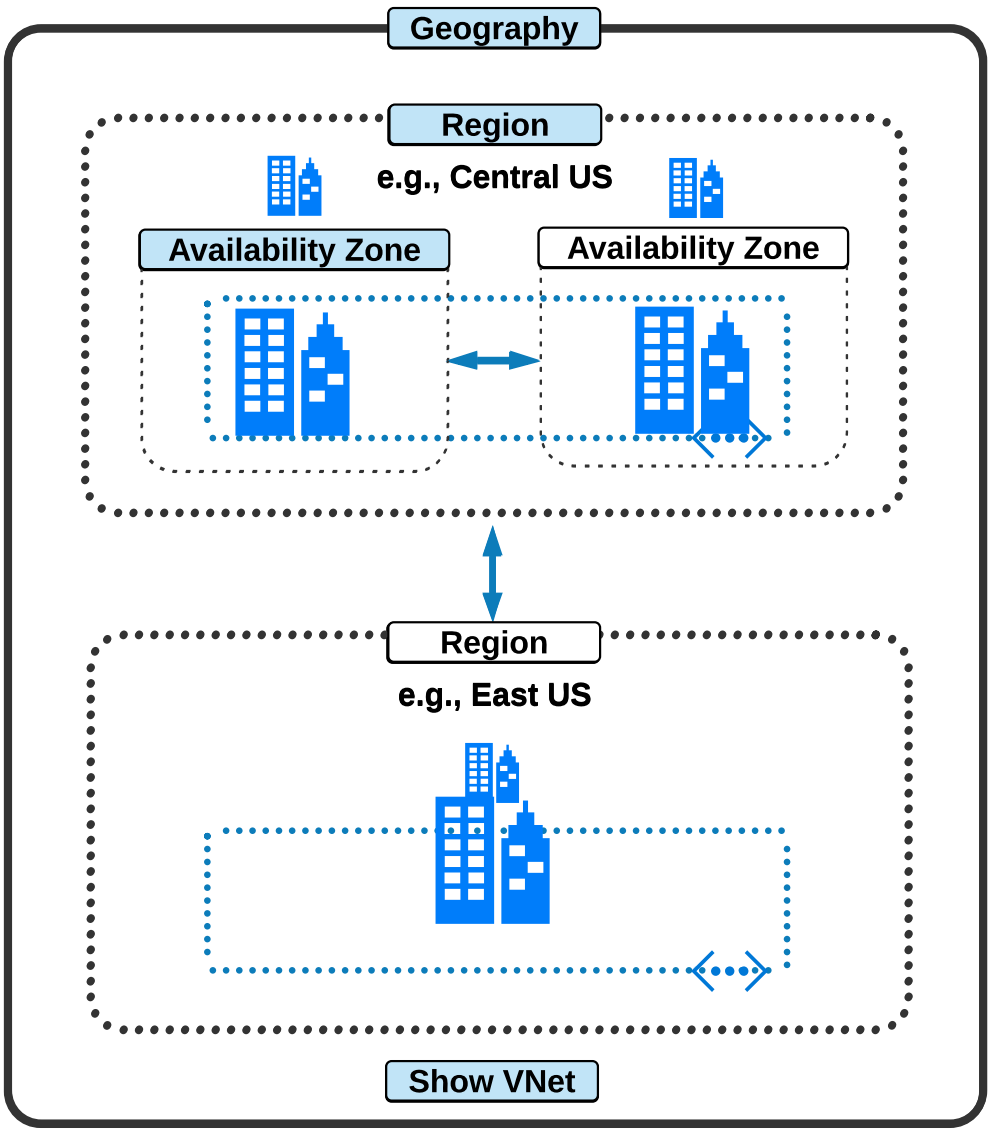
## Physical and Networking
(Virtual Networks)

Virtual networks (VNets) are private logical spaces you can create to isolate network traffic within Azure.

VNets do not span regions, but they do span Availability Zones. More information is available in the Networking page.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

## Physical and Networking
(Virtual Networks)

Virtual networks (VNets) are private logical spaces you can create to isolate network traffic within Azure.

VNets do not span regions, but they do span Availability Zones. More information is available in the Networking page.

Go Back

Azure AD Tenant

Azure Users
(production subscription)

Subscription Owner

Users

Azure Global Infrastructure

Geography

Region

e.g., Central US

Availability Zone

Availability Zone

Region

e.g., East US

Show VNet

On-Premises

Hybrid Cloud

## Geography

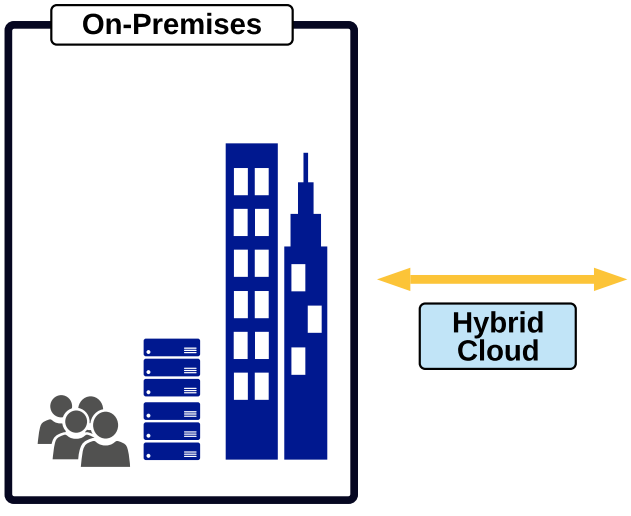
Close

Regions are grouped into geographies. These typically represent data and compliance boundaries, so that Azure customers know what legal, sovereignty, and resiliency requirements are honored.

For example, all regions within the Americas geography will:

- Have data stored at rest within the U.S.
- Comply with international, regional, and industry-specific legislation requirements
- Be available to all

Appendix



## Regions

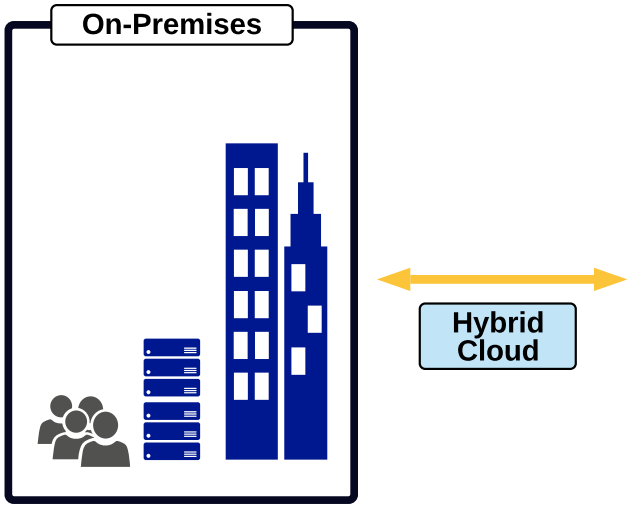
Close

A region represents one or more data centers, which are used to provide Azure services. In terms of physical infrastructure, the region (or location) is what you will use most frequently when deploying services.

For example, you can deploy resources to Australia East, US West, Southeast Asia, and many more.

Data centers within a region are connected through dedicated regional low-latency networks.

Appendix



## Availability Zones

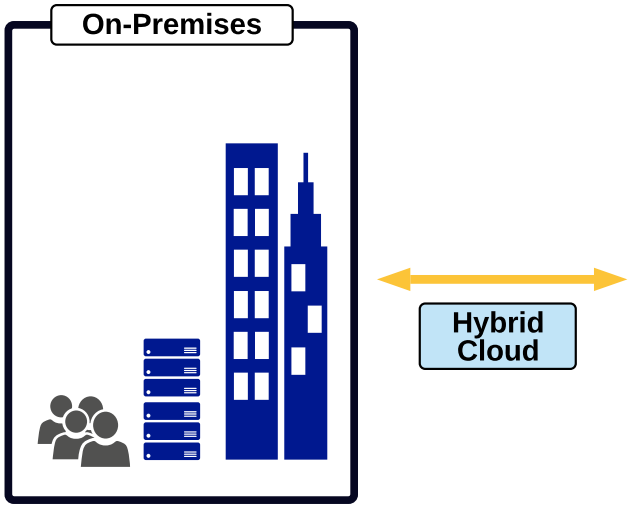
Close

Availability Zones (AZs) are physically separate locations that exist *within* an Azure region.

An AZ is comprised of one or more data centers — each using power, cooling, and networking that is independent of one another.

For more information on how AZs are used in practice, refer to the VM High Availability section found here.

Appendix



# The Blueshift Guide

Subscription and Services Layer

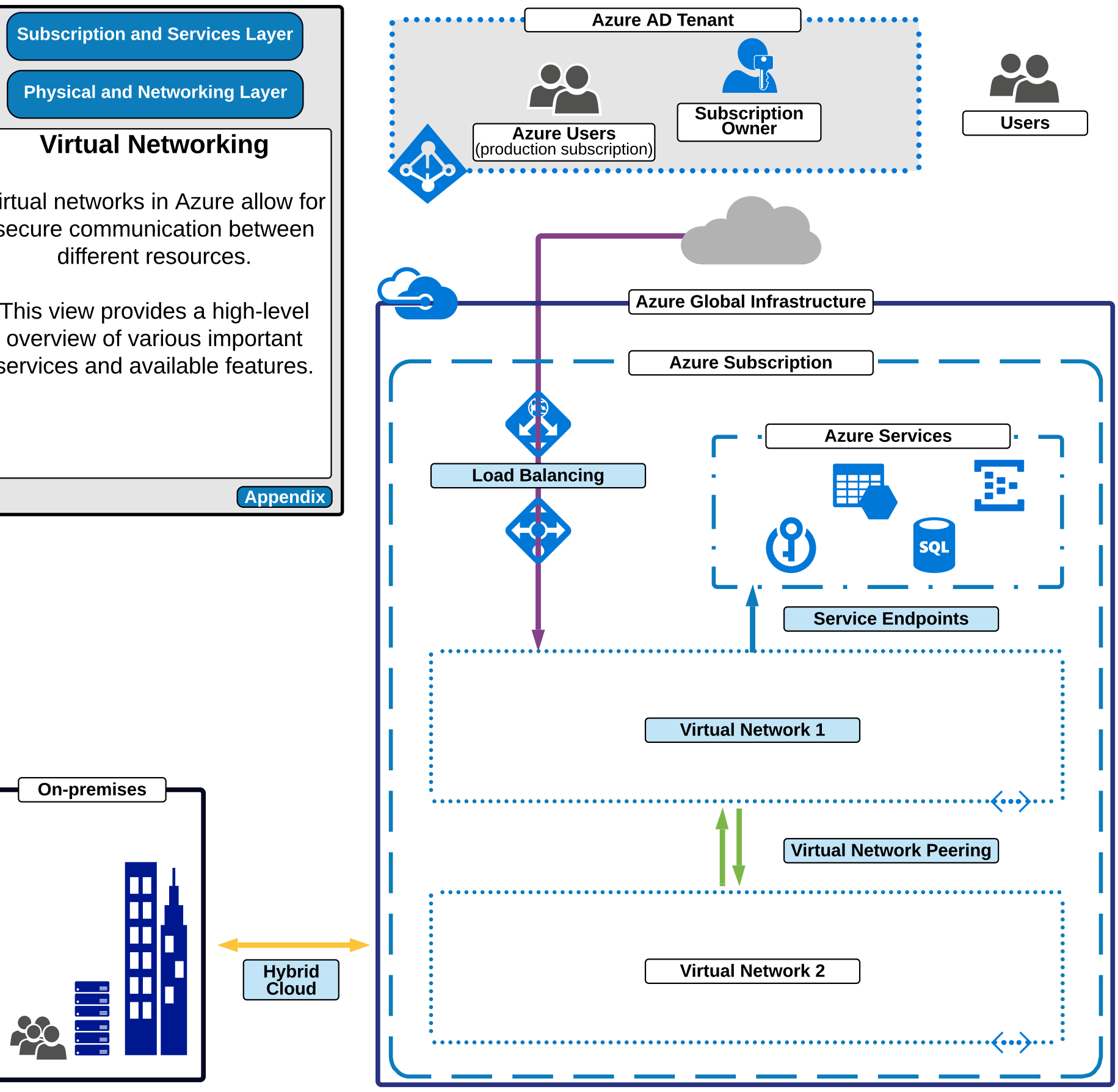
Physical and Networking Layer

## Virtual Networking

Virtual networks in Azure allow for secure communication between different resources.

This view provides a high-level overview of various important services and available features.

Appendix



Subscription and Services Layer

Physical and Networking Layer

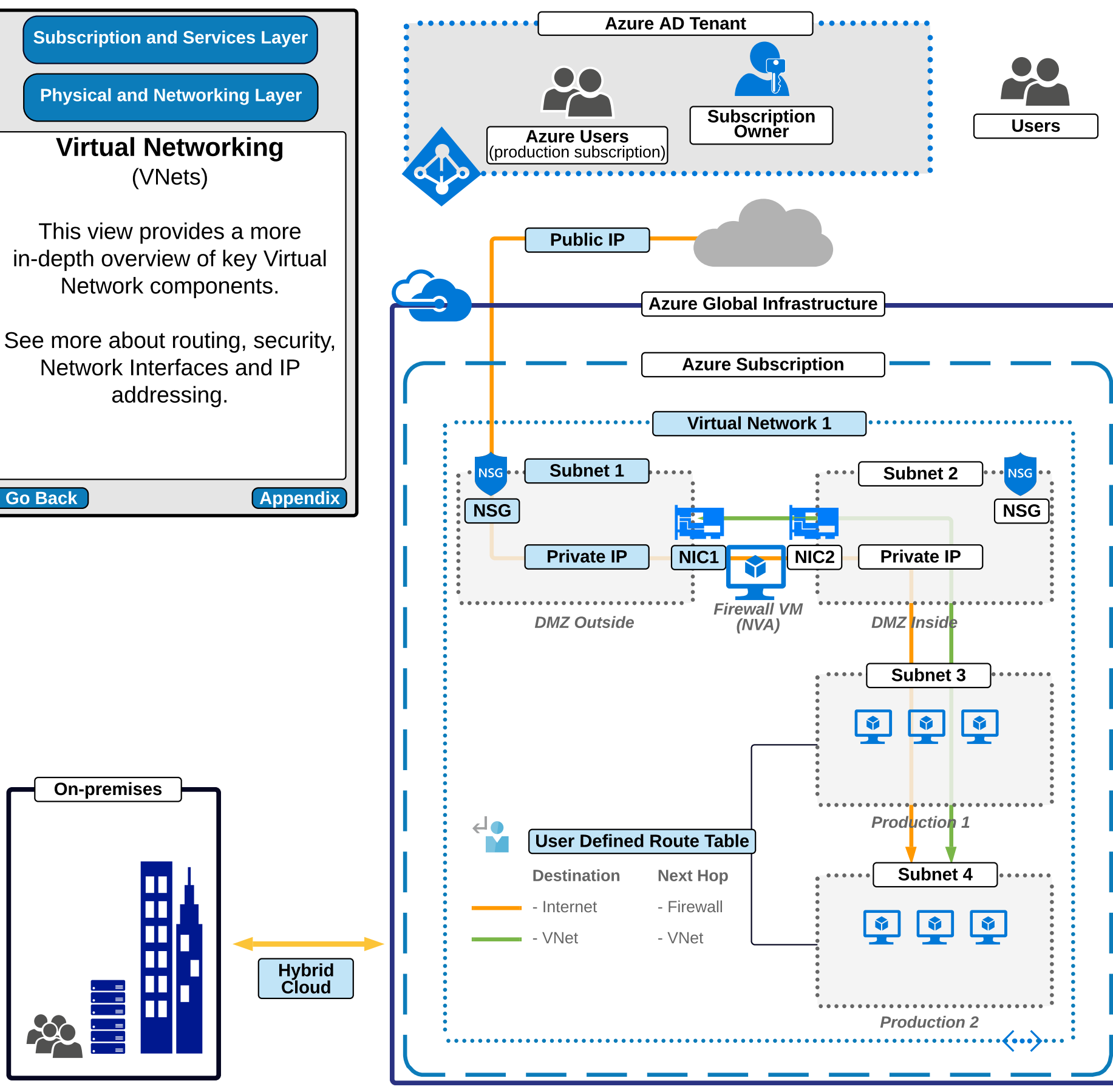
## Virtual Networking
(VNets)

This view provides a more in-depth overview of key Virtual Network components.

See more about routing, security, Network Interfaces and IP addressing.

Go Back

Appendix



Subscription and Services Layer

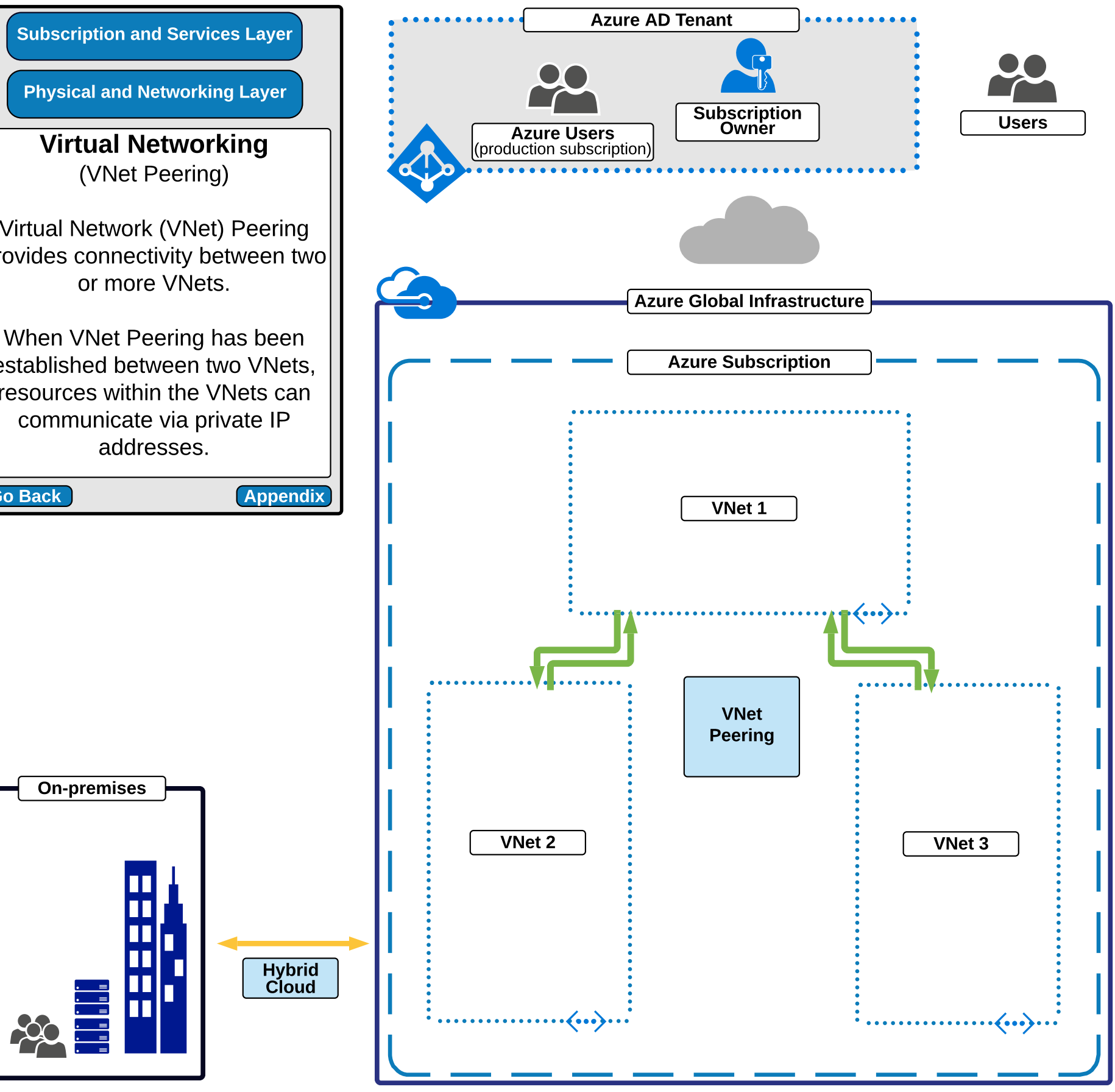
Physical and Networking Layer

## Virtual Networking
(VNet Peering)

Virtual Network (VNet) Peering provides connectivity between two or more VNets.

When VNet Peering has been established between two VNets, resources within the VNets can communicate via private IP addresses.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

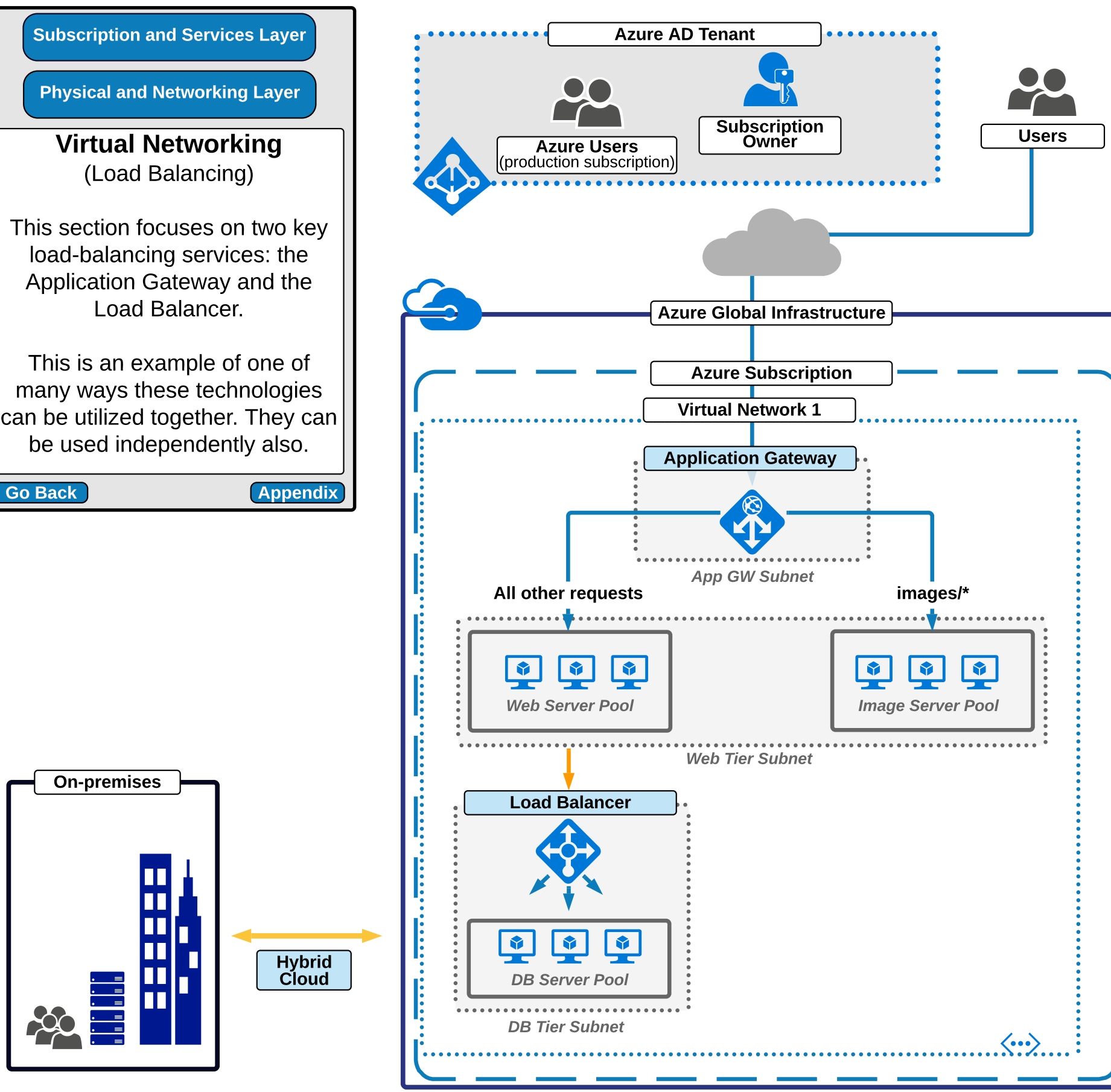
## Virtual Networking
(Load Balancing)

This section focuses on two key load-balancing services: the Application Gateway and the Load Balancer.

This is an example of one of many ways these technologies can be utilized together. They can be used independently also.

Go Back

Appendix



Subscription and Services Layer

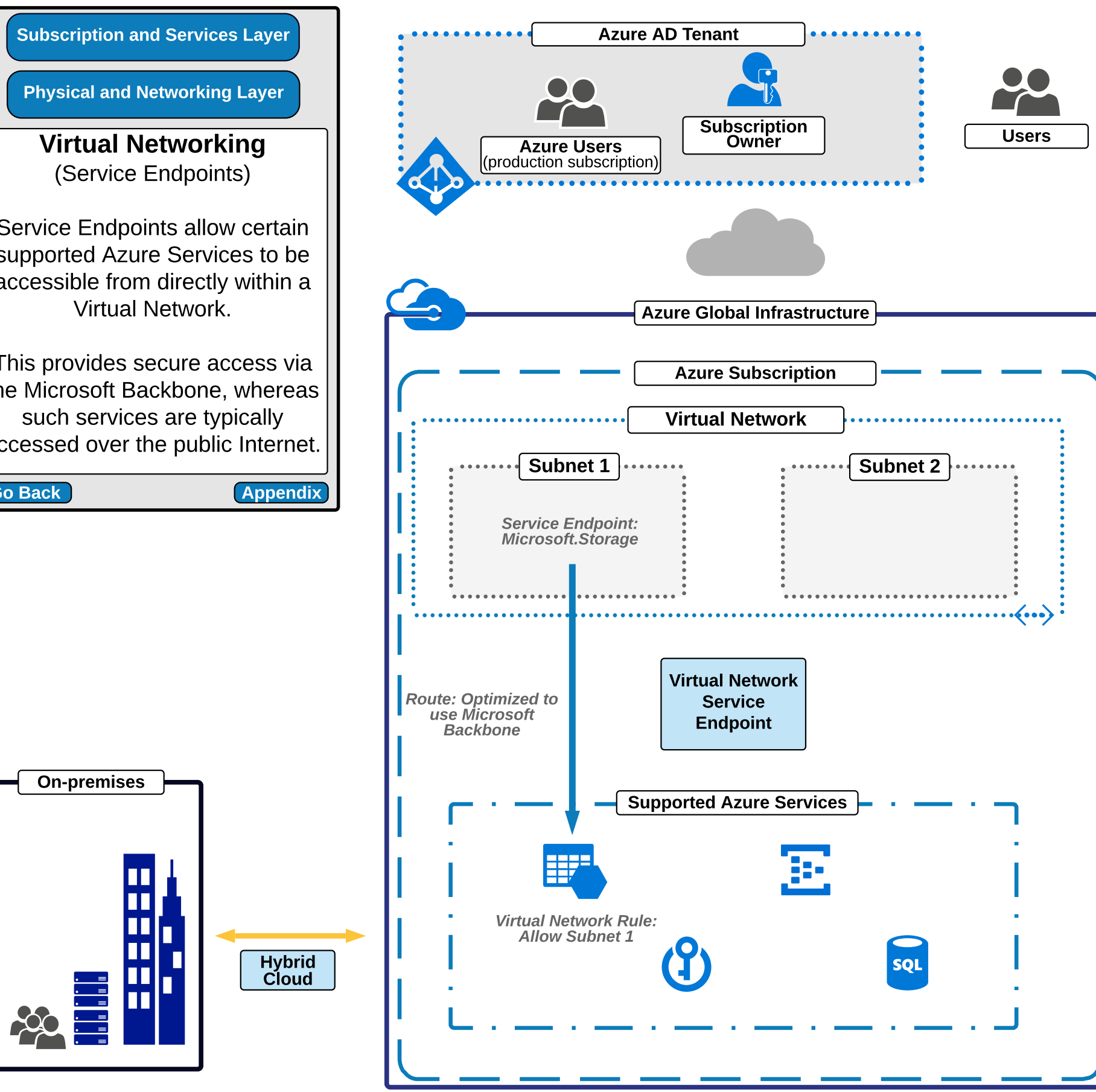
Physical and Networking Layer

## Virtual Networking
(Service Endpoints)

Service Endpoints allow certain supported Azure Services to be accessible from directly within a Virtual Network.

This provides secure access via the Microsoft Backbone, whereas such services are typically accessed over the public Internet.

Go Back
Appendix



Close

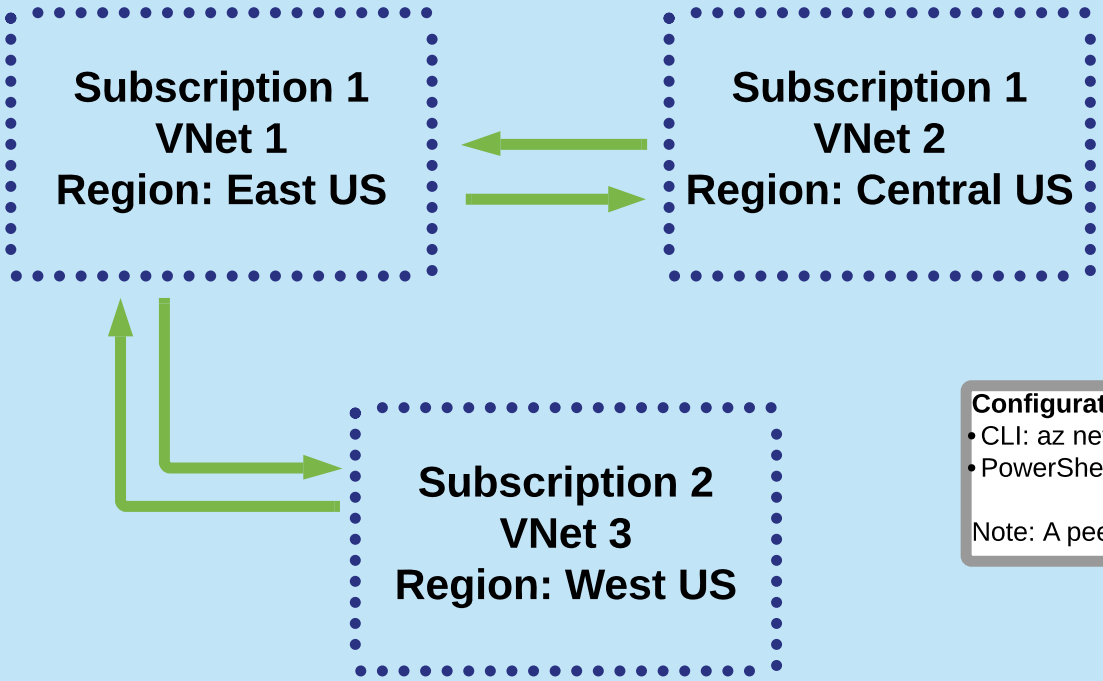
# Virtual Network Peering

Peering allows virtual networks to be connected, which allows resources inside the separate networks to communicate as if they are on the same network.

Traffic flows between the networks over Microsoft's backbone infrastructure, allowing for faster connectivity and better security.

**Use cases:**
- Hub and Spoke topology supports the isolation of workloads, and sharing of common resources

**Key things to remember about VNet Peering:**
- Low latency traffic
- Supports cross-subscription connectivity (using resource ID)
- Supports cross-region connectivity (Global VNet Peering)
- Allows communication between VNets via private IP addresses (public IP addresses are not required)
- Does not support peering between VNets with the same address spaces
- Does not support transitive routing (e.g. VNet 2 cannot communicate with VNet 3 via VNet 1 peering)




**Configuration**
- CLI: az network vnet peering create
- PowerShell: Add-AzureRmVirtualNetworkPeering

Note: A peer must be configured for both directions.

# Service Endpoints

Close

Service Endpoints allow your Virtual Network (VNet) to integrate with a number of Azure services. Configuring a Service Endpoint allows resources in that VNet to communicate privately with the connected Azure service. This provides a secure and optimal route, always traversing the Mitcrosoft Azure backbone.

**Use cases:**

- A processing server (VM) sorts and matches Private Health Information stored in Blob storage. To maintain confidentially, information must not traverse the public Internet.

**Key information:**

- Not all services support Service Endpoints; this list is constantly expanding (see more information here).
- Service Endpoints must be enabled for a specific Azure service, on one or more subnets of a VNet.
- Once a Service Endpoint is enabled, a number of Default Routes will be activated, providing an optimal path between the subnet and the Azure service.
- Service Endpoints can be combined with Virtual Network Rules for the given service, in order to further restrict traffic to the service so that only "allowed" IP addresses and/or VNets are permitted.

Close

# Subnets

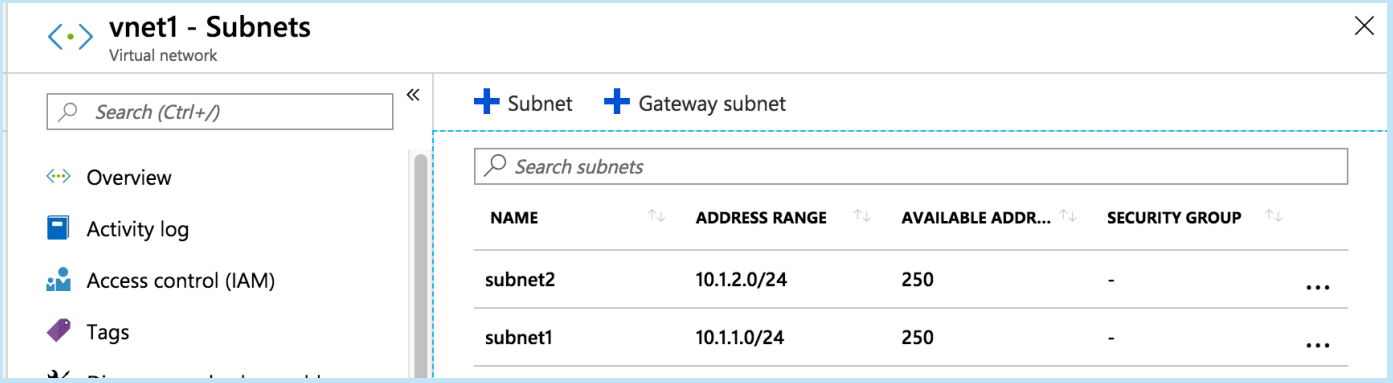
A subnet is used within a Virtual Network (VNet) to help provide isolation and segmentation of workloads. Any resource you deploy to an Azure VNet must be deployed to a subnet within that VNet.

**Use cases:**

- Isolating workloads:
  - You may choose to place all web servers for a specific application within their own dedicated subnet

**Key information:**

- Subnets within the same VNet can communicate to one another by default when they're first created.
- Each subnet is assigned an address space which must be within the address space of the VNet.
- The address spaces of subnets within a VNet cannot overlap.
- Network Security Groups (NSGs) can be assigned to a subnet, to enforce security for traffic on that subnet.
- User Defined Routes (UDRs) can be assigned to a subnet to control routing for traffic on that subnet.
- Azure reserves the first and last IP, along with the x.x.x.1 - x.x.x.3 addresses of each subnet.



Close

# Network Security Groups

Network Security Groups (NSGs) are used to provide network layer security for resources within a Virtual Network (VNet). When attached to a resource, they can allow or deny traffic based on rules you configure.

**Key information:**

- Best practice is to block ALL traffic, except that which is required, sometimes known as "default deny."
- NSGs can be applied to either a Network Interface Card (NIC), a subnet, or both.
- NSG rules are stateful, so reply traffic is allowed automatically regardless of other rules.
- NSGs contain "Default Rules" which cannot be deleted; to override these requires higher priority rules.
- Rules are processed according to priority: the lower the number, the higher the priority.
- Once a rule is matched, no further rules are processed.

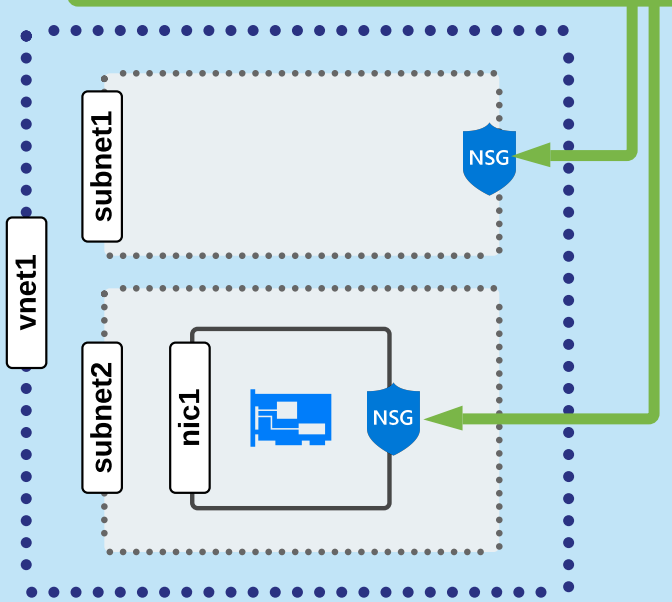
**Configuration**

- Rules must be either inbound or outbound
- A priority must be assigned; the lower the number, the higher the priority
- Ports can be a wildcard, a number, a range, or a comma separated list (e.g. 80,443)
- Protocol can be set to either **Any, TCP,** or **UDP**
- Source/Destination can be an **IP Address, Service Tag, Application Security Group,** or **Any**
- Action must be either Allow or Deny

Note that an NSG will have no effect if it is not assigned to a resource

Note: When an NSG is applied to both a subnet and a NIC, rules on both NSGs will be evaluated. E.g. inbound traffic blocked at the subnet would not reach the NIC.

Consider the NSG rules at each step in the path of the traffic flow.




Inbound security rules

| PRIORITY | NAME | PORT | PROTOCOL | SOURCE | DESTINATION | ACTION |
|---|---|---|---|---|---|---|
| 110 | inbound-allow-rdp | 3389 | Any | 1.2.3.4 | Any | Allow |
| 65000 | AllowVnetInBound | Any | Any | VirtualNetwork | VirtualNetwork | Allow |
| 65001 | AllowAzureLoadBalancerInB... | Any | Any | AzureLoadBal... | Any | Allow |
| 65500 | DenyAllInBound | Any | Any | Any | Any | Deny |

Outbound security rules

| PRIORITY | NAME | PORT | PROTOCOL | SOURCE | DESTINATION | ACTION |
|---|---|---|---|---|---|---|
| 900 | outbound-block-all | Any | Any | Any | Any | Deny |
| 65000 | AllowVnetOutBound | Any | Any | VirtualNetwork | VirtualNetwork | Allow |
| 65001 | AllowInternetOutBound | Any | Any | Any | Internet | Allow |
| 65500 | DenyAllOutBound | Any | Any | Any | Any | Deny |

Close

# Network Interfaces

Network Interfaces (also known as Network Interface Cards, or NICs) are independent resources which, when connected to a Virtual Machine (VM), enable network connectivity with other networked resources.

**Use cases:**

- VM network communication with the Internet, Azure, and even on-premises (via VPN or ExpressRoute)

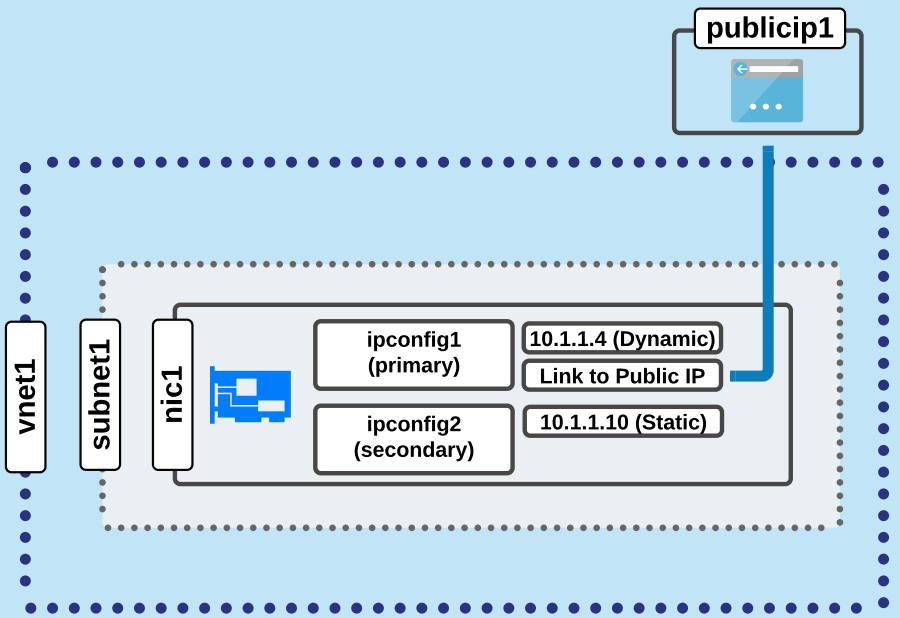
**NICs:**

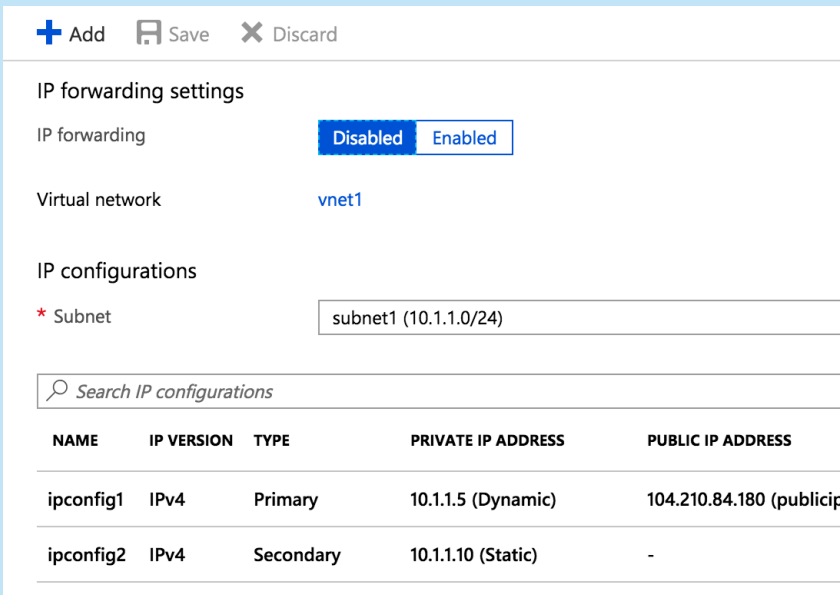
- Are independent resources which can be created separately from a Virtual Machine
- Provide private connectivity using a private IP address on the VNet it is attached to
- Require 'IP Forwarding' to be enabled, if they are to forward packets (e.g. a VM firewall)
- Include at least one "IP configuration," which specifies the properties of Private and Public IP addressing
- Can either have DNS settings specified manually, or inherited from a VNet

**Configuration**

- CLI: az network nic create
- PowerShell: New-AzureRmNetworkInterface

Note: A VM must be created with at least one NIC, and can only have as many NICs as the VM size supports.



Close

# User Defined Routes

User Defined Routes (UDRs) are used so that you can override the default routing behavior of Azure.

**Use cases:**

- Forcing all traffic which is going to the Internet to go via a Virtual Machine (VM) configured as a firewall
  - This is also known as a Network Virtual Appliance, or NVA.
- Blocking access to the Internet

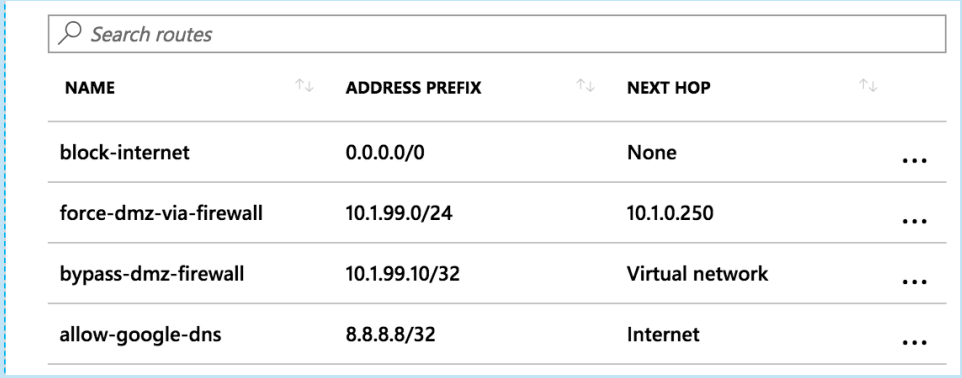
**Key information:**

- The UDR includes a number of routes, essentially defining the path for traffic to take for a given destination
  - When traffic matches a route (according to the Address Prefix) it will route according to the Next Hop.
- The effective route for a resource will be a combination of system routes, UDR routes, and BGP routes
  - The most specific route (known as Longest Prefix Match, or LPM) will take precedence
  - If there are multiple routes with matching LPMs the following priority is used: UDR, BGP, system routes
- UDRs are associated with one or more subnets, but each subnet can only use one UDR

**Configuration**

A UDR is created and associated with a Subnet. The UDR contains one or more Routes. Each Route is defined as below:

- Address Prefix: specifies the destination network using CIDR notation
- Next Hop Type: specifies where the traffic should be routed
  - Virtual Network - routes traffic to locally connected resources (on the VNet)
  - Virtual Network Gateway - routes traffic to the VNet Gateway for this VNet
  - Internet - routes traffic to the Internet
  - Virtual Appliance - routes traffic to a Virtual Machine (IP must be specified)
  - None - drops traffic
- Next Hop Address: used to specify an IP address of a Virtual Appliance



Close

# IP Addressing

For Azure resources to communicate over a network, an IP address is required. There are two types of IP addresses we can configure in Azure - a Private IP address or a Public IP address.

**Private IP addresses** are used to communicate locally within an Azure Virtual Network (VNet), and with on-premises networks when using a VPN or ExpressRoute private connection

- **Public IP addresses** are used when communicating with resources over the Internet

**Key information - Public IP addresses:**

- Independent resources which can be associated with other resources - for example VMs or VPN Gateways
- Can be configured in two main types (referred to as SKUs in Azure) - Basic or Standard
- Basic SKU
  - Supports static or dynamic allocation methods
  - Can be associated with any resource which supports Public IP addresses
- Standard SKU
  - Supports static allocation only
  - Can not be assigned to all resources (currently only supports network interfaces (NICs), public standard Load Balancers, Application Gateways, and VPN Gateways)
  - Includes default *deny* rules, and requires a Network Security Groups to explicitly allow/whitelist traffic
  - Supports newer high availability functionality (e.g. Zone Redundancy)

**Key information - Private IP addresses:**

- A configuration item of various resources (e.g. VM NICs, Internal Load Balancers, Application Gateways)
- Different from a Public IP address, which is itself a resource
- Supports both Dynamic and Static allocation

***Note:*** The first four addresses in each subnet are reserved and cannot be assigned as a Private IPs

**Configuration**

- CLI: az network public-ip create
- PowerShell: New-AzureRmPublicIpAddress

***Note:*** Private IP addressing is typically assigned in-line as part of the creation of various resources (e.g. a Load Balancer Front End Pool).

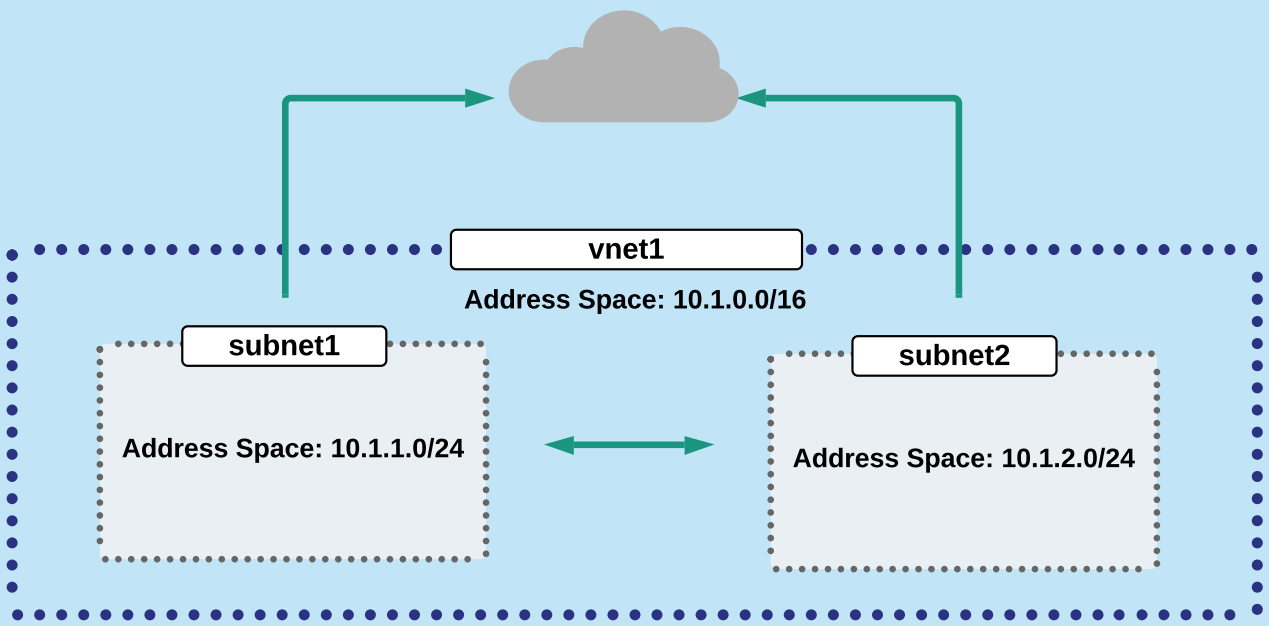
Close

# Virtual Networks (VNet)

Virtual Networks (VNets) are used for create a virtual private network within Azure, where resources can be networked to one another similar to a private on-premises environment.

**Key information:**

- The VNet has an address space, for example 10.1.0.0/16
- Resources connect to subnets within a resource to gain network access
- Subnets within the VNet must exist within the same address space
- Note it is possible to have multiple address spaces for a VNet
- VNets can be connected to one another, and on-premises, permitted the address spaces do not overlap
- There are limitations with VNets, such as no DHCP, multicast, broadcast, or encapsulated packets



Close

# Application Gateway - Overview

Overview | Configuration

Azure Application Gateway is a load balancer which for web traffic. This allows distribution of traffic, in a similar fashion to the Azure Load Balancer, but with additional Layer 7 (http/https) functionality.

Load balancing at layer 7 means that the Application Gateway can provide additional functionality (compared to the Load Balancer) such as, URL-based routing, SSL termination, web application firewall functionality, session affinity, and more.

**Key information about the Application Gateway:**

- Provides layer 7 load balancing functionality (URL-based routing, SSL termination, etc.)
- A number of features are currently "In Preview" and do not currently appear on the exam
- The Application Gateway can either be Standard or WAF tier:
  - WAF (web application firewall) tier includes protection against common web application exploits.
- Using the current generation of Application Gateways requires manual sizing and high availability:
  - Increasing instance count to two or more helps achieve high availability of your Application Gateway,
  - Use a SKU size of medium or large, for production environments.
- Using version 2 of the Application Gateway provides a somewhat more automated experience:
  - High availability can be automated,
  - Availability Zones can be used,
  - SKU size is no longer required to be defined during provisioning.

**Important limitations to be aware of:**

- Only one public IP address is supported on an Application Gateway
- Public IP addresses cannot be static (unless using V2 SKU); for this reason a CNAME DNS entry is recommended to be pointed to the DNS address of the Application Gateway
- The same port (e.g. port 80) cannot be used for both public and private listeners
- Rules are processed in the order they are listed (ensure path based rules appear before basic rules)

# Application Gateway - Configuration

Close

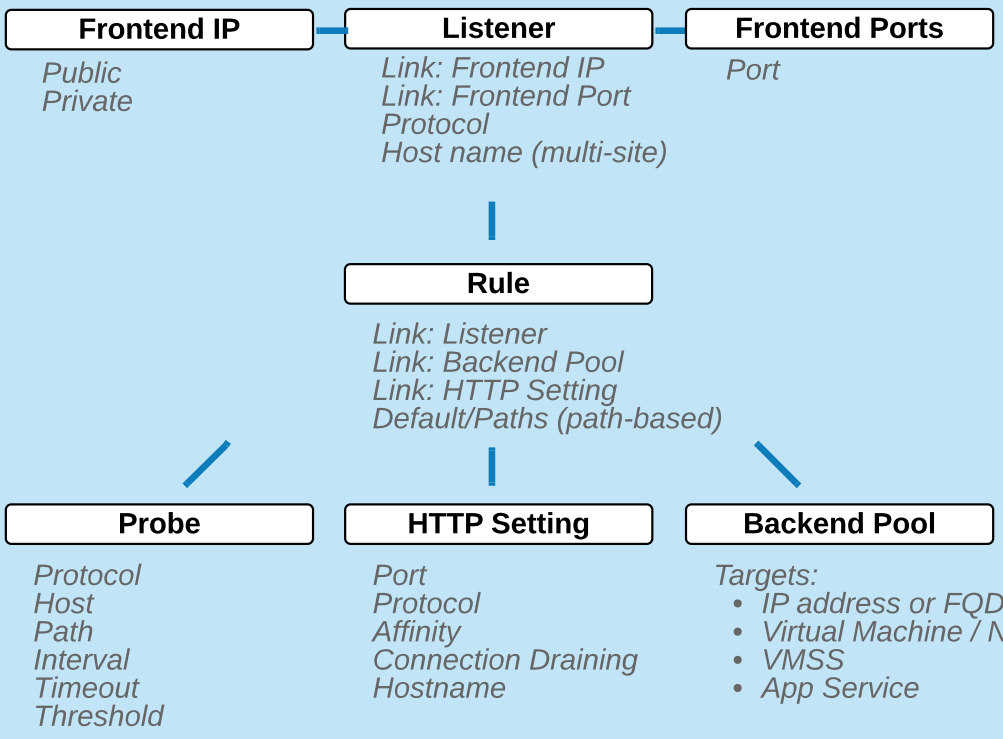
Overview | Configuration

Since the Application Gateway provides more functionality than the Load Balancer, the configuration is more detailed and versatile.

Below is an overview of the various resource properties, and how they tie together to create the overall functioning configuration of the Application Gateway.

**Important configuration notes:**

- The Application Gateway must exist in a dedicated subnet which cannot be shared by any other resources, other than additional Application Gateways.
- Rules are processed in the order they are configured; multi-site rules should appear first.




**Configuration Items:**

**Frontend IP:** The IP address of the Application Gateway: supports both public and private IP addressing

**Frontend Port:** The port that the Application Gateway listens on

**Listener:** Combination of a protocol, frontend IP, and frontend port.

- Multi-site listeners allow the specification of a host name.

**Backend Pool:** IP addresses/FQDN's (any that are reachable), or NICs belonging to resources that are hosting the web application

**Custom Health Probe:** Used to determine whether the backend member is healthy

**HTTP setting:** Configures how traffic is routed to backend members (including port, protocol, cookie-based-affinity, probe, timeout)

**Rule:** Ties all configuration items together, allowing the definition of an inbound port mapping

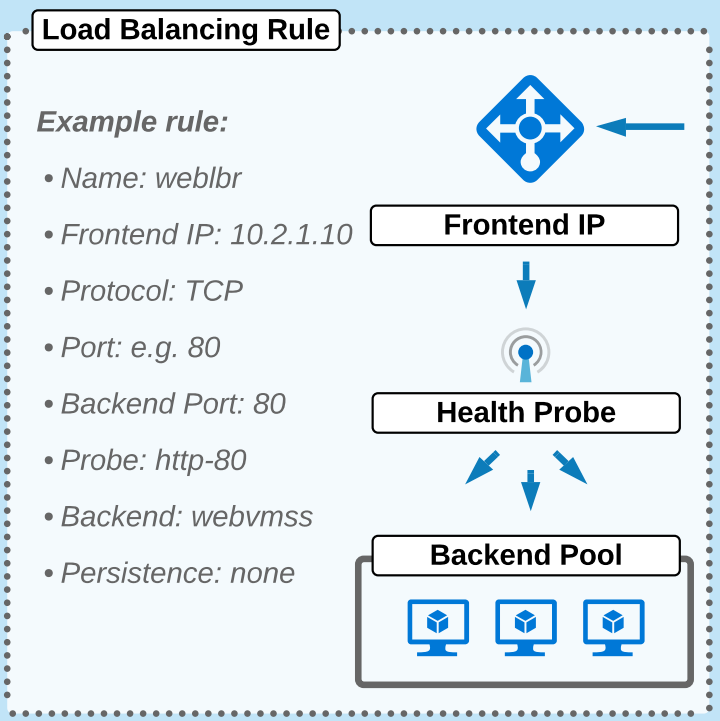
# Load Balancer

Close

Load Balancers provide the ability to distribute traffic amongst multiple resources, helping to support high availability and elasticity. An Internal Load Balancer is only accessible to resources on or connected to an Azure Virtual Network (VNet). Public Load Balancers, on the other hand, are accessible over the Internet.

**Key information about the Load Balancer:**

- Azure Load Balancer supports the distribution of traffic at layer 4 (TCP, UDP).
- Session affinity / persistence can be configured to ensure a client requesting traffic through the Load Balancer gets a response from the same backend VM, based on Client IP, or Client IP + Protocol.
- Load Balancers can be either Standard or Baskic SKU, some of the key differences are as follows:
  - Backend pool size: Standard SKU supports up to 1000 instances, basic SKU supports up to 100
  - Backend pool endpoints: Standard SKU adds the ability to blend types (VMs, Availability Sets, VMSS)
  - Health probes: Standard SKU adds HTTPS
  - Availability Zones: Standard SKU supports zone-redundant and zonal frontends for inbound/outbound
  - Outbound connections: Standard SKU adds greater control over outbound connectivity




**Configuration Items:**

**Frontend IP:** The IP address of the Load Balancer itself

**Health Probe:** Used for determining whether the final destination instance is available to receive traffic

**Backend Pool:** Defines the ultimate destination for traffic

**Load Balancing Rule:** Ties all configuration items together, allowing the definition of an inbound port mapping

**Inbound NAT Rules:** Provides the ability to map inbound traffic directly to instances behind the Load Balancer.

Close

## Coming Soon

The AZ-300 course is released **in preview**, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

To continue, please click the close (X) button.

Subscription and Services Layer

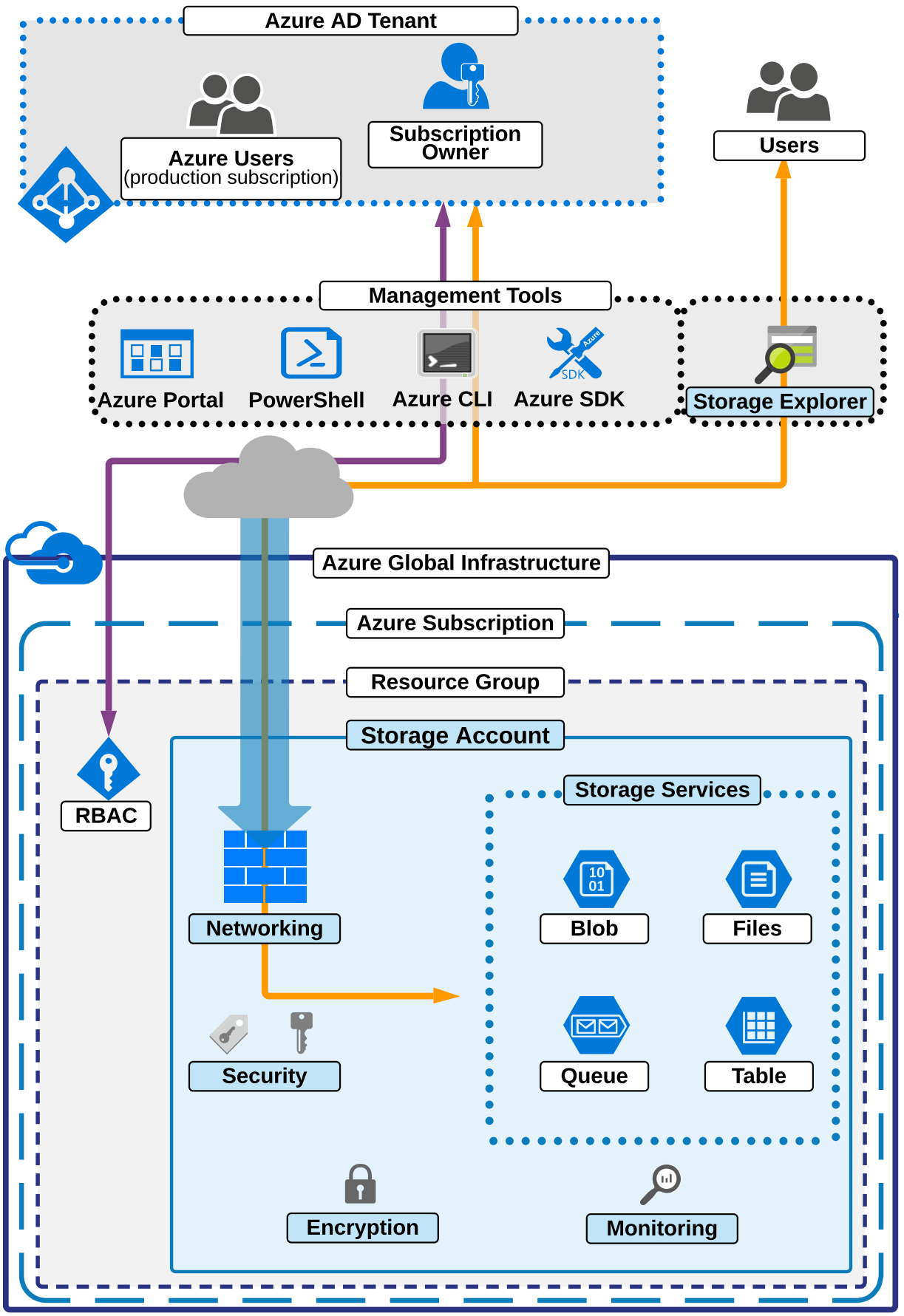
Physical and Networking Layer

## Azure Storage

Azure Storage represents a range of services used for housing data. At the root of these services is the Azure Storage Account.

Storage Accounts can contain:
Blob: Object based storage
Files: Managed file sharing
Queues: Message queues
Table: NoSQL key-value data

Appendix



Subscription and Services Layer

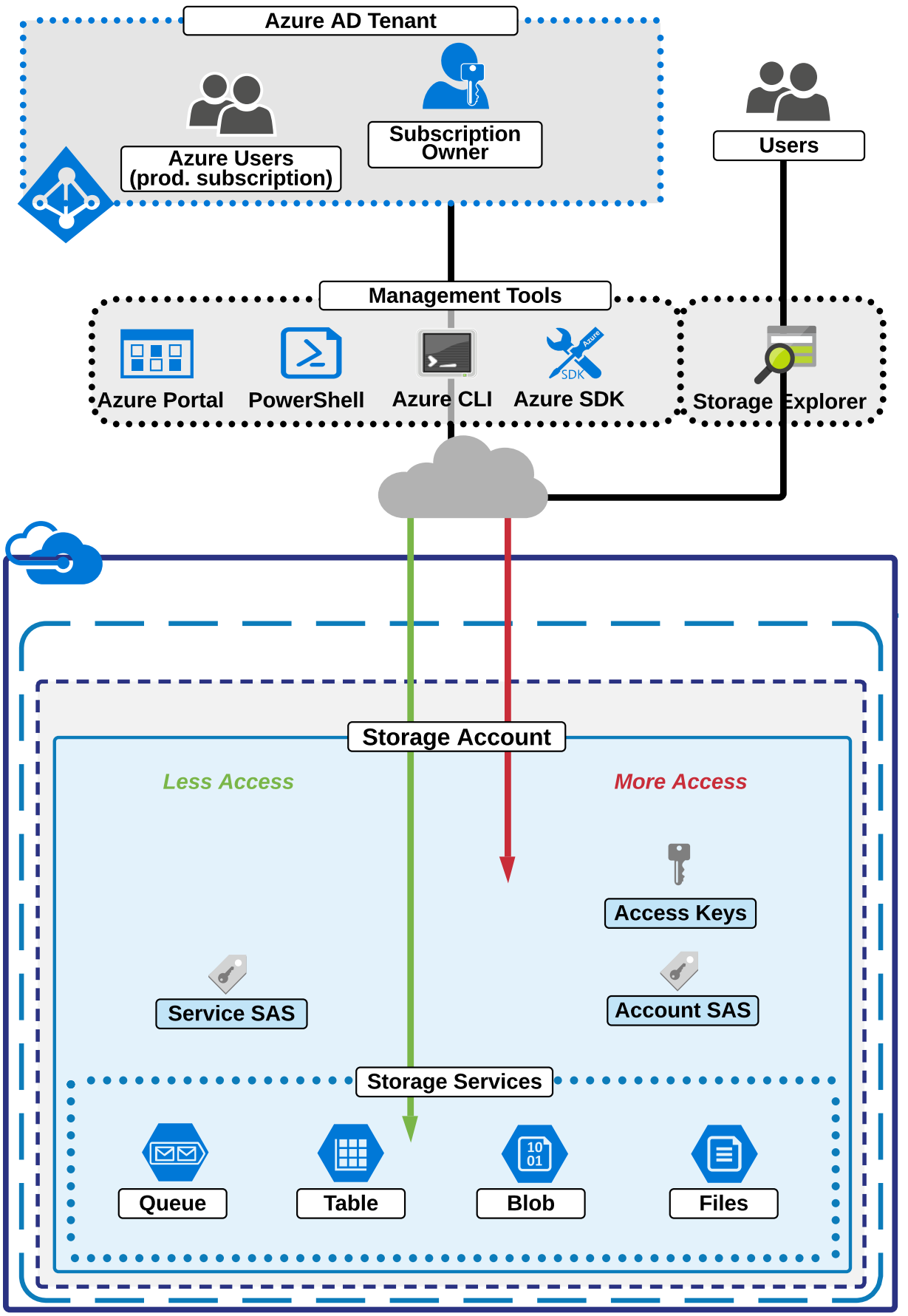
Physical and Networking Layer

## Storage Accounts
(Access Control)

This page provides an overview of the primary methods available to secure Storage Accounts at the data layer.

These methods can be used in combination with network layer security. The outline is in the Storage Networking page.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

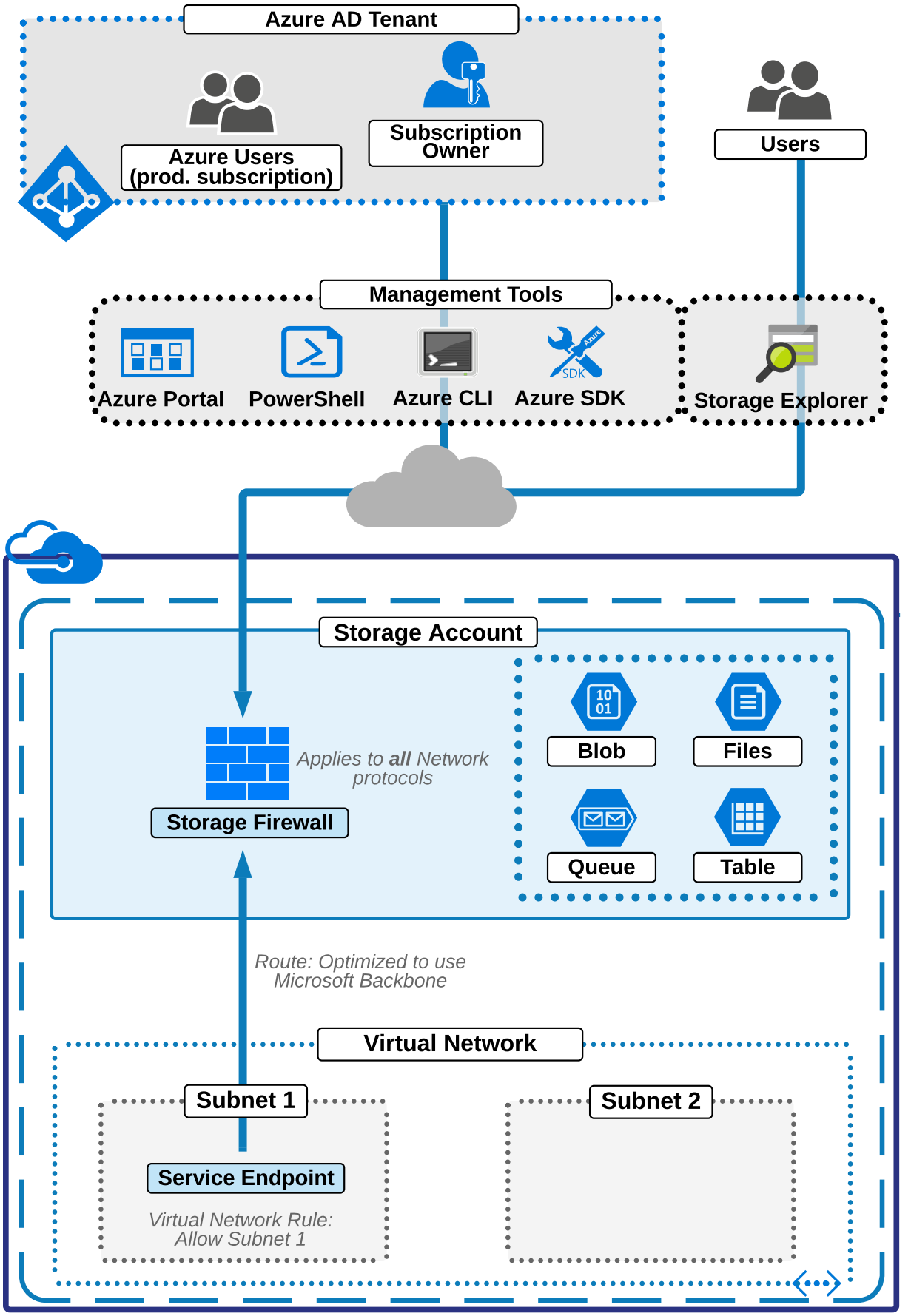
## Storage Accounts
(Networking)

Storage Accounts are designed to be accessed over the Public Internet.

This page details key features which help secure Storage Accounts at the network layer.

Go Back

Appendix



Subscription and Services Layer

Physical and Networking Layer

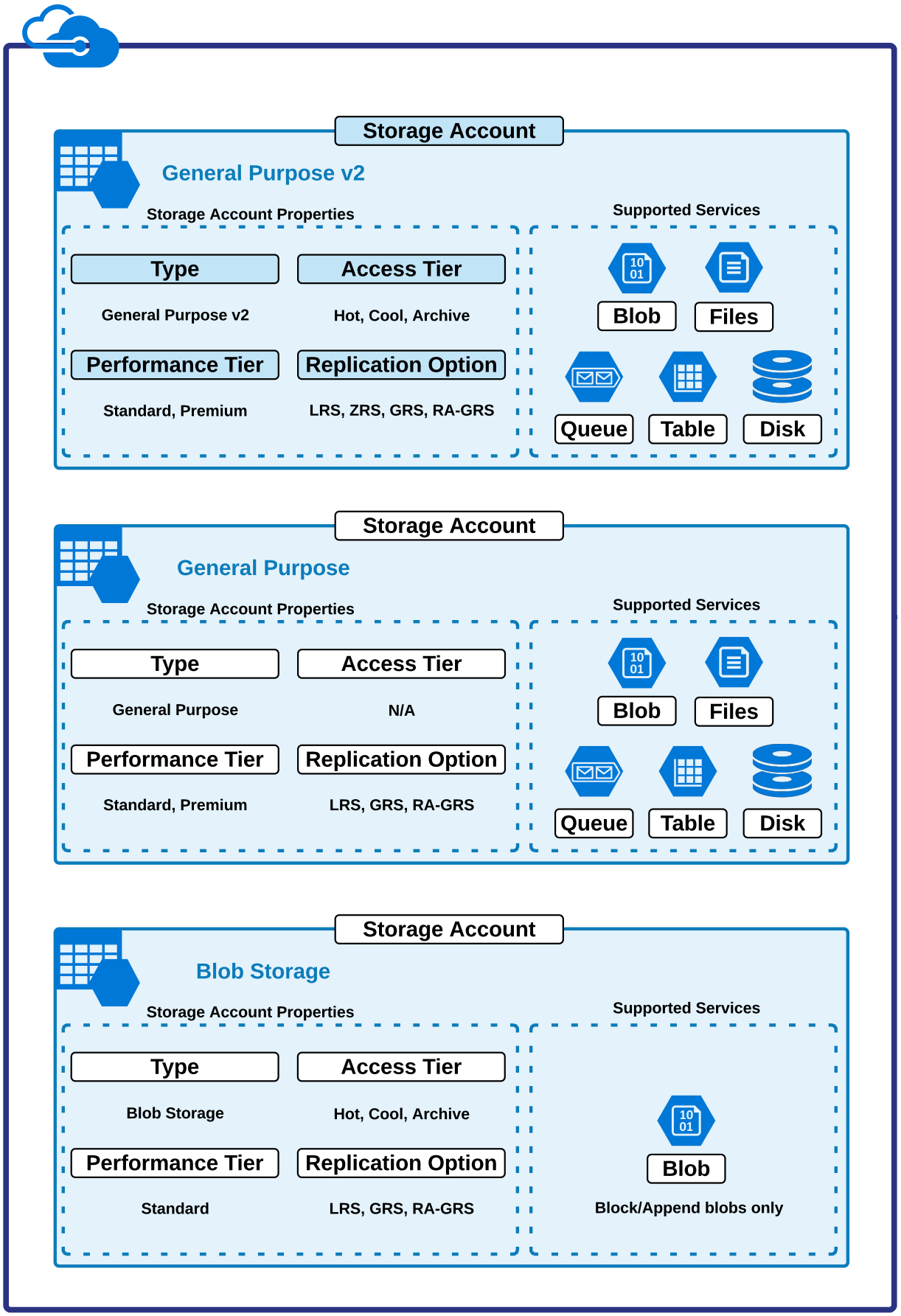
## Storage Accounts
(Detailed View)

All Blobs, Files, Queues, Tables, and Disks are created within Storage Accounts.

This page helps to illustrate key configuration properties, and important differences with the three types of Storage Accounts.

Go Back

Appendix



Close

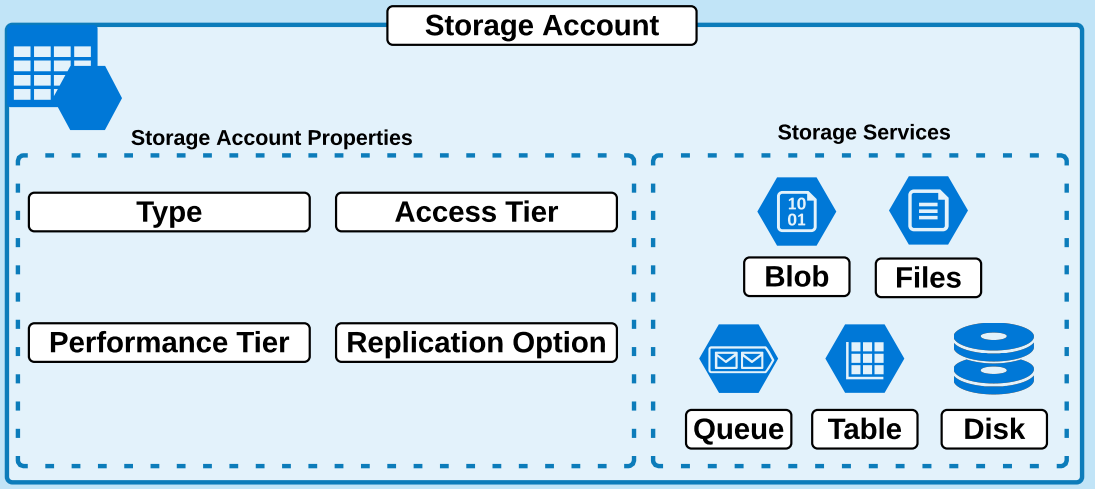
# Storage Accounts (SA)

Storage Accounts (SA) are the parent container of the primary Azure Storage objects: Blobs, Files, Queues, and Tables. Before you create any of these services, you must create an SA.

A number of important configuration properties are defined at the SA level, which directly influences the configuration of the contained storage services. Examples are replication, performance, and access tiers.

**Key information:**

- Storage Accounts must be uniquely named across Azure (for ALL of Azure, not just your subscription).
- Three types are available: general-purpose v2 (GPv2), general-purpose (GP), and blob storage (Blob).
- There are limitations and variations between all of the three SA types. Refer to the previous SA diagram
  - For example, Cool and Archive access tiers are only available with GPv2 and Blob SAs
- All SAs are encrypted at rest using Storage Service Encryption. This cannot be disabled.



# Storage Account - Type

Close

There are three types of Storage Accounts (SA) you can create, and these directly influence the features, pricing, and properties available to you. The three SA types are:
1. Blob storage
2. General-purpose (GPv1)
3. General-purpose v2 (GPv2)

**Blob storage**
- Special type of SA which only supports the creation of Blob storage (and only block and append blobs)
- Originally (prior to GPv2) required in order to leverage the additional access tiers

**General-purpose v1**
- Standard type of Storage Account which supports all storage services, now superseded by GPv2
- Provides lowest transaction prices, but higher storage prices
- Supports Azure Service Manager (Classic)

**General-purpose v2**
- Supports all storage services, and provides access to the latest features
- Supports Access Tiers (Cool and Archive storage)
- Supports zone-redundant storage
- Recommended for most scenarios

| Storage Account Type | Supported Services | Supported Performance Tiers | Supported Access Tiers | Replication Options |
|---|---|---|---|---|
| General-purpose v2 | Blob, File, Queue, Table, and Disk | Standard, Premium | Hot, Cool, Archive | LRS, ZRS, GRS, RA-GRS |
| General-purpose v1 | Blob, File, Queue, Table, and Disk | Standard, Premium | N/A | LRS, GRS, RA-GRS |
| Blob storage | Blob (block and append blobs only) | Standard | Hot, Cool, Archive | LRS, GRS, RA-GRS |

Close

# Storage Account - Access Tier

Microsoft provides three Access Tiers for Storage Accounts. This ultimately helps optimize the pricing model of your ***blob storage***, based on the frequency of reads and writes. The three options are:

- Hot storage: for storing *frequently accessed data*
- Cool storage: for storing *infrequently accessed data* which is *stored for at least 30 days*
- Archive storage: for storing *rarely accessed data* which is *stored for at least 180 days*

**Use cases:**

- Hot storage: the typical use case of data which is actively read from and written to frequently
- Cool storage: often used for short-term backup, or large volumes of infrequently used (e.g. media) content
- Archive storage: commonly used for long-term backups, or for data which must be retained for long-term in order to comply with law/regulation (e.g. financial records going back 7 years)

**Key information:**

- Configuring the Access Tier at the SA level will set the default; this can still be set at the object level.
- Archive tier can only be set at the object level, not as a default at the SA level.
- General Purpose v1 (GPv1) type Storage Accounts do not support Access Tiers.

Close

# Storage Account - Performance Tiers

There are two Performance Tiers available to Storage Accounts (SA):

1. Standard: Storage Accounts which are backed by magnetic drives and provide low cost storage
2. Premium: Storage Accounts backed by solid state drives with high performance, low-latency access

**Use cases:**

- Premium Performance Storage Accounts can currently only be used with Azure Virtual Machine (VM) disks
  - They are suitable for database servers and other high I/O applications
- VMs which use Premium storage also qualify for a 99.9% uptime/connectivity SLA, even when running as a single instance. This is compared to 99.99% for two VMs running in an Availability Set.
- Standard Performance Storage Accounts are currently used for the majority of use cases, as they are supported by all Storage Account storage services.

**Key information:**

- Premium Performance Access Tier is only supported by General Purpose type SAs.
- Currently, creating an SA with the Premium Performance tier SA will result in the following limitations:
  - Only Virtual Machine disks can be stored in the SA
  - Only Page Blobs will be able to be created in the SA
  - Only locally redundant storage (LRS) will be able to configured for the SA

Close

# Storage Account - Replication Options

To ensure that data within Storage Accounts is always available, it is possible to configure Replication Options. This controls how and where your data is replicated. There are four available options:

1. Locally-redundant storage (LRS): Replicates data to another storage scale unit
2. Zone-redundant storage (ZRS): Replicates data across three storage clusters within a single region
3. Geo-redundant storage (GRS): Replicates data to a separate region
4. Read-access GRS (RA-GRS): GRS with the added ability of being able to read from the secondary copy

**Important information about how data is synchronized:**

- LRS and ZRS replication occurs synchronously
- GRS and RA-GRS replication occurs asynchronously

| Scenario | LRS | ZRS | GRS | RA-GRS |
|---|---|---|---|---|
| Node failure within a datacentrer | Yes | Yes | Yes | Yes |
| Entire datacenter failure | No | Yes | Yes | Yes |
| Region-wide outage | No | No | Yes | Yes |
| Read access to your data in the event of region-wide outage | No | No | No | Yes |

More information can be found in the following Microsoft article:
https://azure.microsoft.com/en-us/support/legal/sla/storage/v1_3/

Close

# Storage Explorer

Azure Storage Explorer is an application which can be used to manage the contents of Storage Accounts. It is more than a simple "explorer," providing not just the ability to read and write data, but also the ability to configure various Storage Account properties.

Storage Explorer can be installed on Windows, macOS and Linux, and is also available (in preview) through the Azure Portal.

**Key information about Storage Explorer:**

- View, edit, and create blobs, files, queues, and tables
- Manage configuration items for the various resources
- Configure access control such as Shared Access Signatures

# Shared Access Signatures

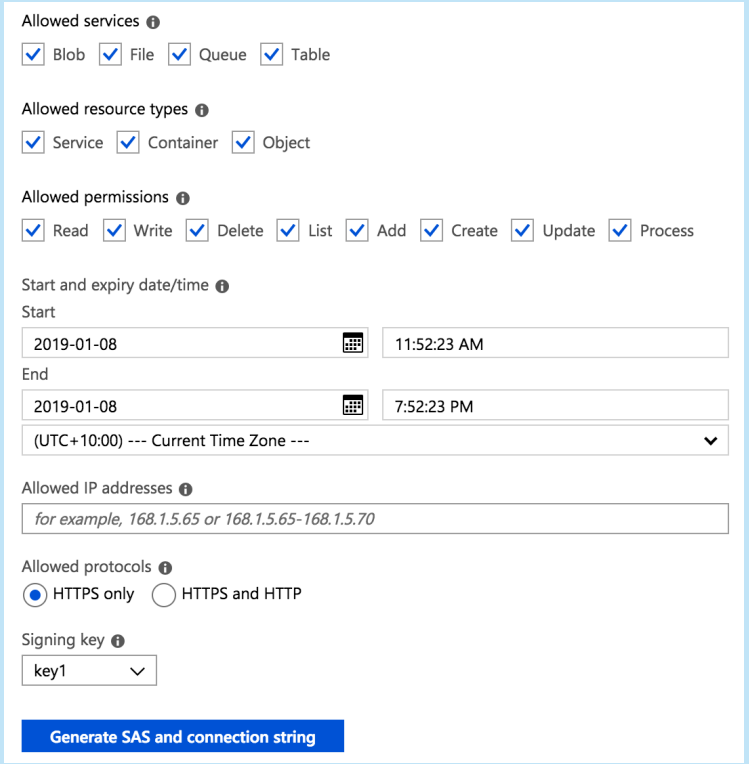
Close

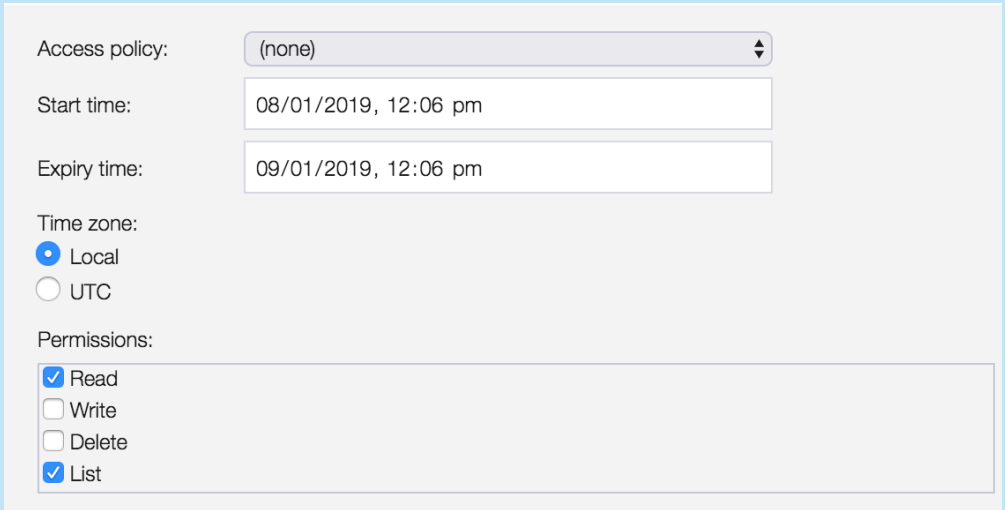
Shared Access Signatures (SAS) are one method of controlling access to objects within a Storage Account (SA). There are two types of SAS: Service SAS, and Account SAS.

The Account SAS can provide access to one or more resources within an SA, whereas the Service SAS provides access to a resource in just one of the storage services (blob, file, queue, table).

**Key info:**

- An SAS is a URI which can provide access to resources; anyone who is given the URI will gain access.
- An SAS can include start/expiry times, permissions/operations allowed, IP and protocol restrictions.
- All SAS URIs contain a signature; these are constructed from the SAS parameters to provide authorization.
- The Service and Account SAS URIs are similar, however Account SAS specifies additional parameters.
- Service SAS only provides access to an individual resource.
- Account SAS can provide access to one or more resources, as well as additional operations.
- Refer to this Microsoft article for a detailed breakdown.
- Access signatures can be constructed using code, or most management tools, including Storage Explorer.



Close

# Storage Account Access Keys

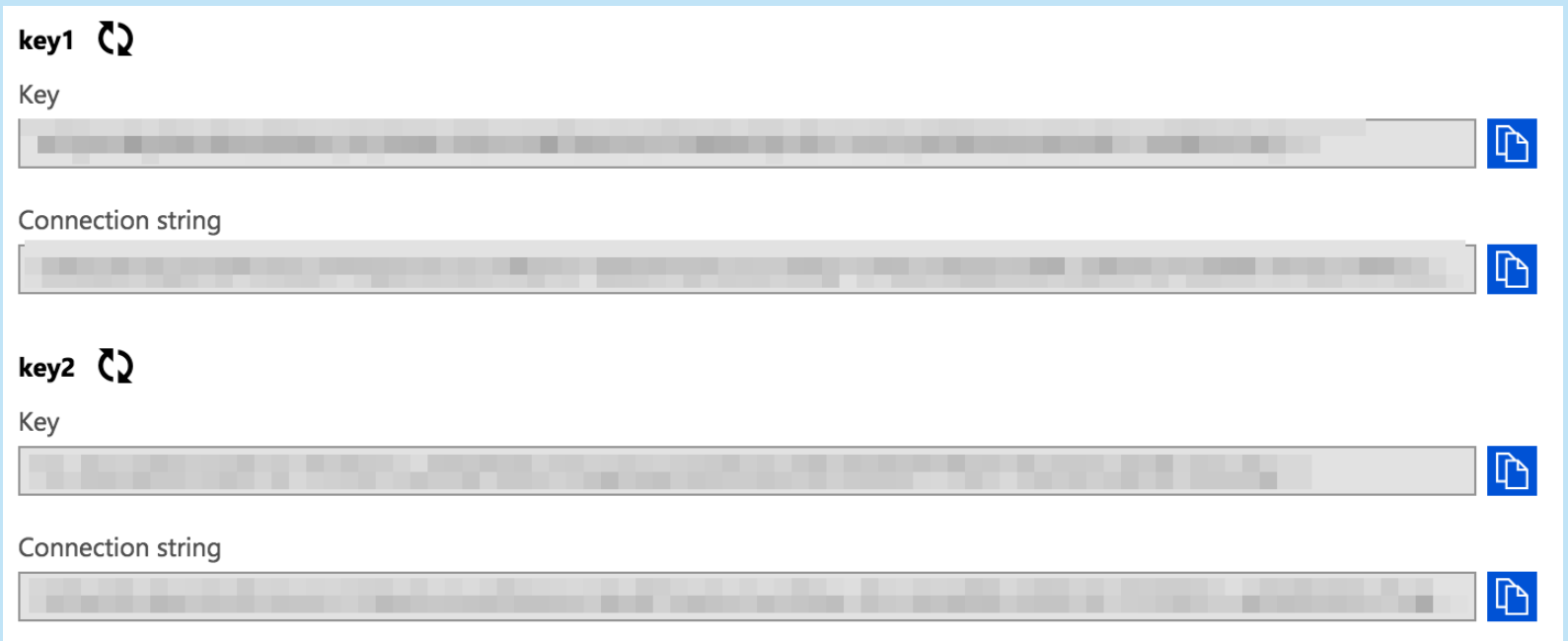
Storage Account (SA) Access Keys are automatically generated during the creation of any SA. Specifically, Azure will generate two : *key1* and *key2*. Both keys provide full and complete access to the SA.

**Use cases:**

- SA Access Keys should only be used sparingly, either for development or test purposes, or when no other forms of access control can be utilized.

**Key information about Storage Account Access Keys:**

- They are automatically generated keys which provide complete access to a Storage Account
- Access Keys can be regenerated:
  - Regenerating an Access Key will result in a brand new key being created.
  - When an Access Key is regenerated, the current key becomes immediately unusable.
  - Access Keys can be regenerated one at a time, ensuring there is always at least one valid key.



# Storage Account - Network Access

Close

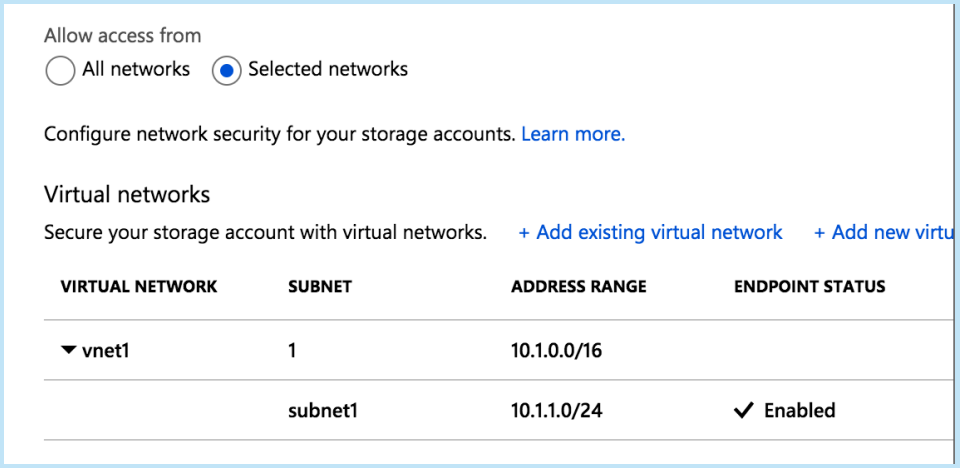
There are two key components of Storage Account (SA) network access:
1. Virtual Network Service Endpoint
2. Storage Firewall

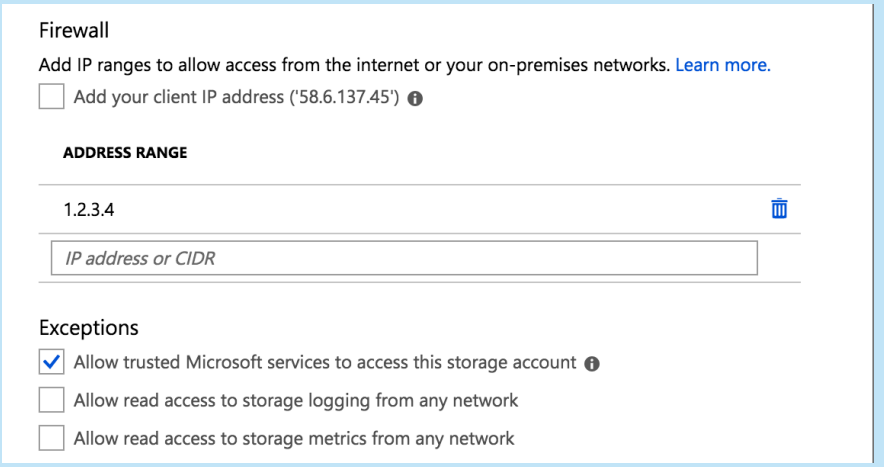
Virtual Network (VNet) Service Endpoints are responsible for private connectivity between your VNets and SAs. Service Endpoints can be used for other services - you can find more information in the Networking section.

Storage Firewalls provide network level access controls, and can be used independently, or in conjunction with Service Endpoints. They provide the ability to allow and deny SA access based on IP addressing.

**Key information:**

- Storage Firewall rules apply to ***all*** network protocols (including REST and SMB).
- By default, Storage Accounts are accessed over the public Internet.
- Enabling the firewall creates a DEFAULT DENY rule which will block all access to a Storage Account.
- Whitelisted IP addresses and Exceptions can be enabled to grant access.
- Only Public IP addresses can be added to network rules; VNets can be used for granting private access.
- Exceptions can be enabled to grant Microsoft services and common types of access (logging/metrics).



Close

# Storage Encryption

Azure Storage provides a number of features to secure data using encryption. The security strategy used is different for each of the Azure storage services (blobs, files, queues, tables).

**Key information about Azure Storage encryption:**

- Encryption at rest is enabled for all Storage Accounts, and cannot be disabled:
  - The encryption service is referred to as Azure Storage Service Encryption (SSE)
  - SSE uses Microsoft managed keys by default, but can use customer managed keys
- Client-side encryption can be used for encryption in transit, for example:
  - HTTPS for transferring data in/out of Azure Storage (enforceable with Storage Access Signatures)
  - Azure Files requiring SMB 3.0 which encrypts during transit with Azure file shares
  - Using the Azure Storage Client Library support for client-side encryption
- Azure Key vault can be used to support both encryption in transit and at rest, by managing encryption keys

**Where can I find more information?**

https://docs.microsoft.com/en-us/azure/storage/common/storage-security-guide
https://docs.microsoft.com/en-us/azure/security/security-storage-overview

Close

# Azure Storage Services

Azure Storage services, which can be created within an Azure Storage Account, are as follows:

Blobs: Massively scalable object-based storage for test and binary data
Files: Managed file shares, similar in intention to that of an on-premises file server
Queues: Messaging solution used for decoupled apps, built according to a message-based architecture
Tables: NoSQL storage solution for schemaless key/attribute based data

More information can be found at this Microsoft article, here.

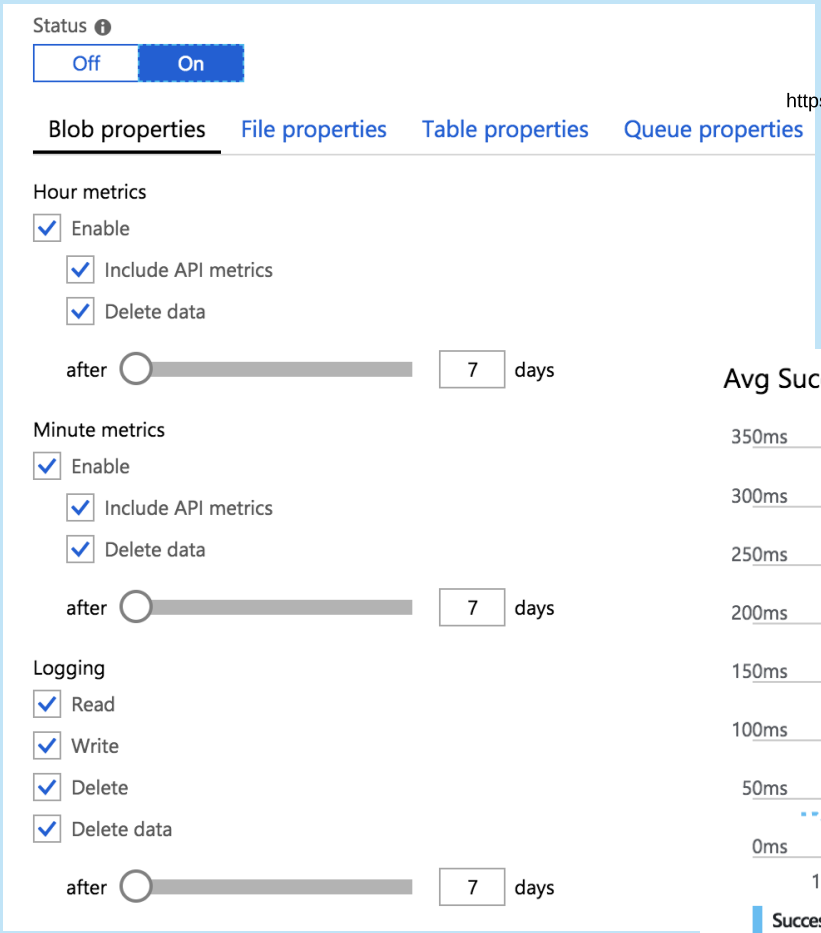
Close

# Storage Monitoring and Analytics

Azure provides the ability to monitor and analyze various aspects of storage. This includes performance metrics, and the activity log.

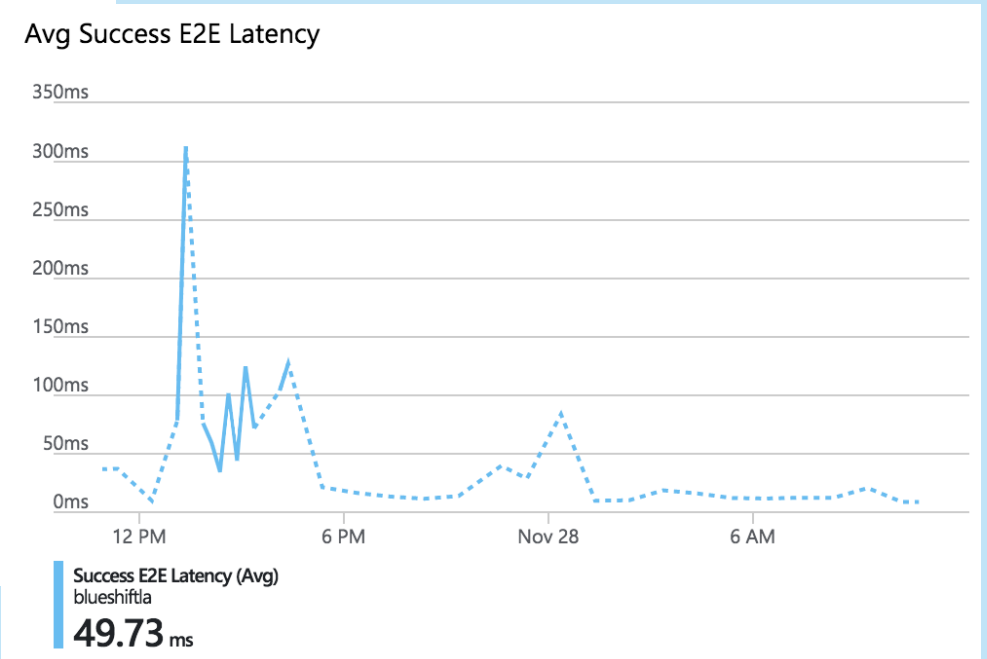
More information can be found in the Monitoring section.

## Enable Logs Per Service




https://www.dropbox.com/s/wmlkyxqb0il9qf0/run_script.mkv?dl=0

## View Service Metrics



Subscription and Services Layer

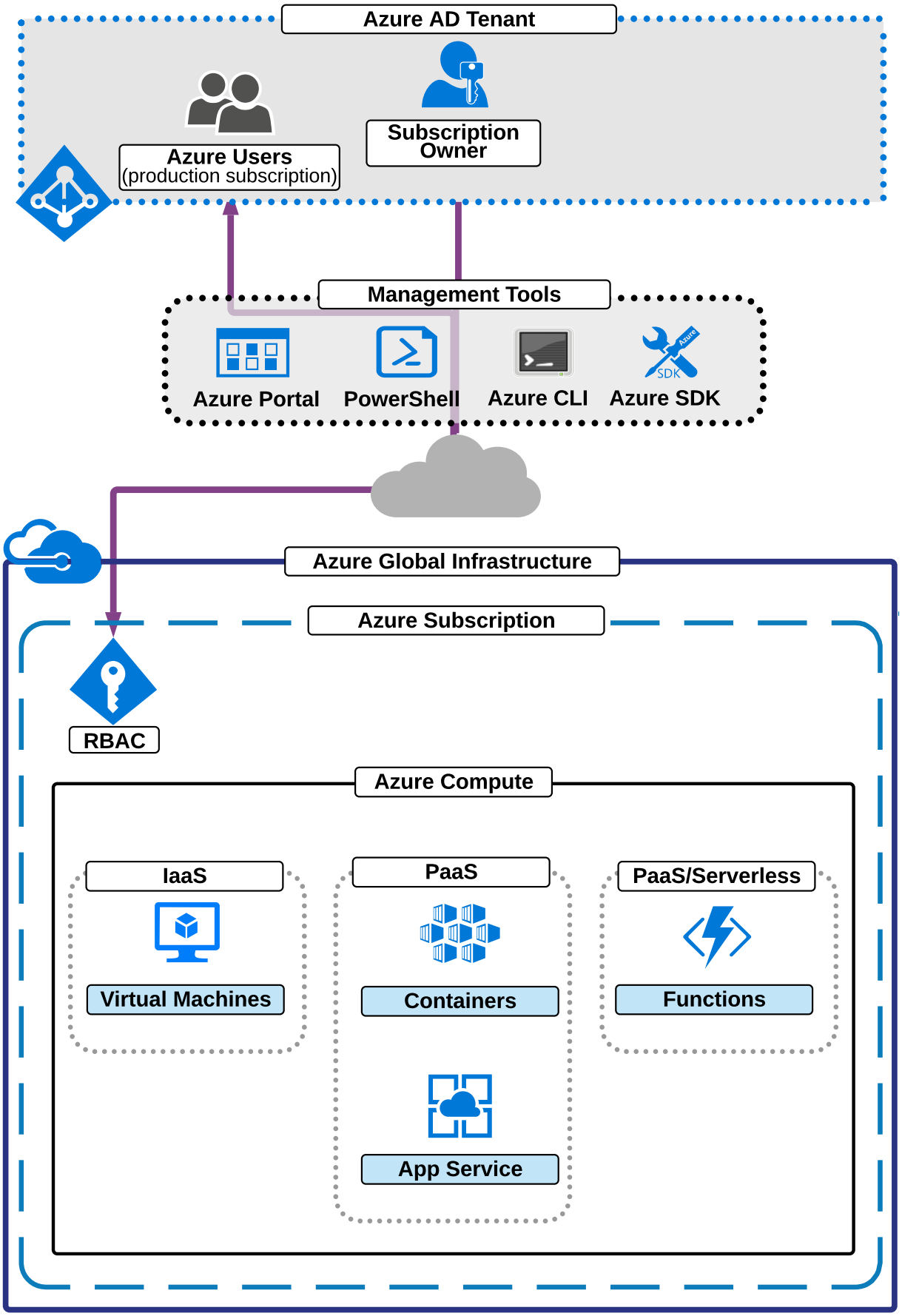
Physical and Networking Layer

## Azure Compute Services

Microsoft provides a range of hosting models to represent the various services for hosting applications in Azure.

Of these services, Virtual Machines are the most common. However as more cloud-native development occurs, this is changing.

Appendix



Subscription and Services Layer

Physical and Networking Layer

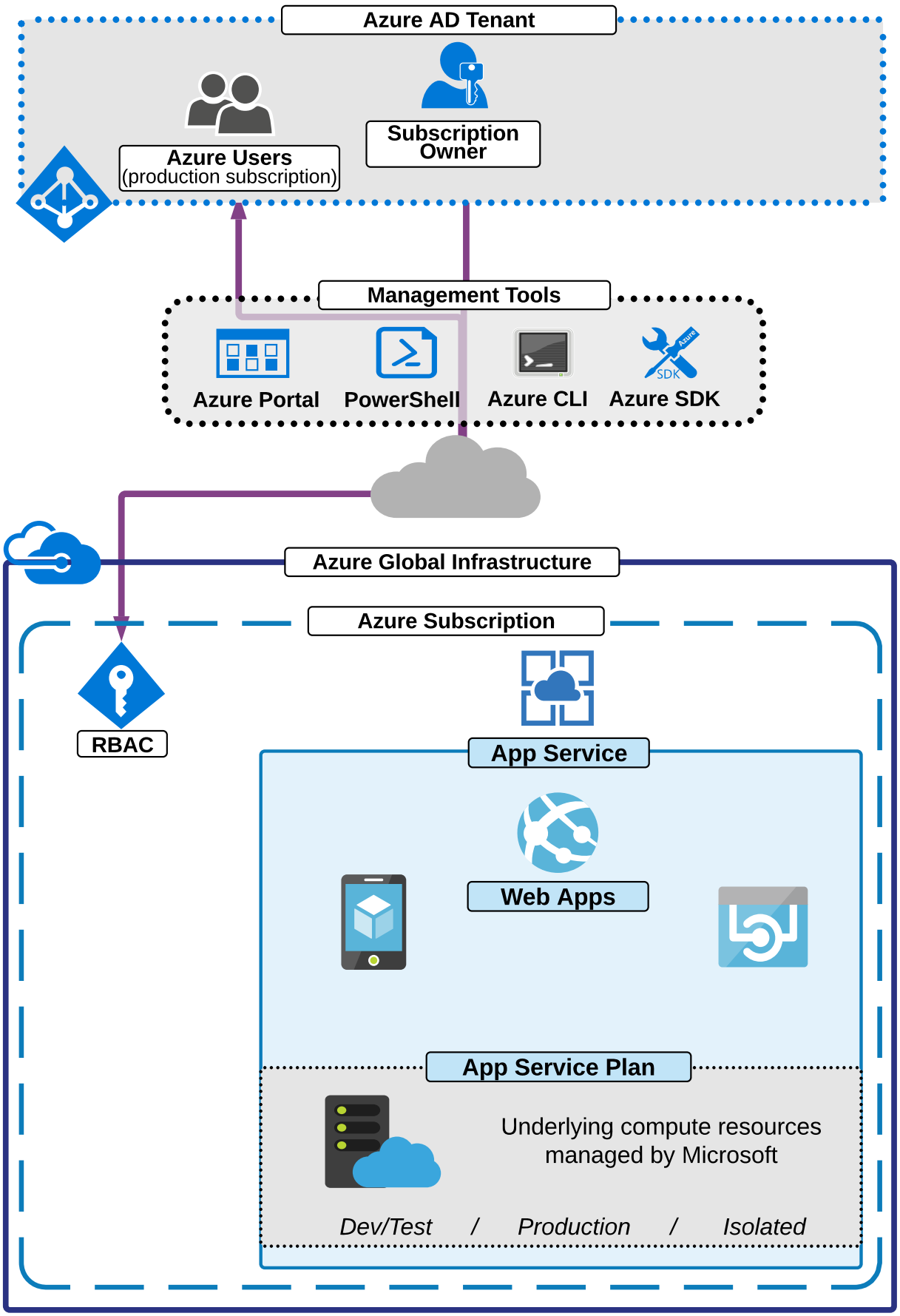
## Azure App Service

Azure App Service is the underlying service which powers Web Apps, Mobile Apps, and API Apps within Azure.

The underlying compute resources, and included features for a hosted app, are determined by the App Service Plan.

Go Back

Appendix



Subscription and Services Layer

Physical and Networking Layer

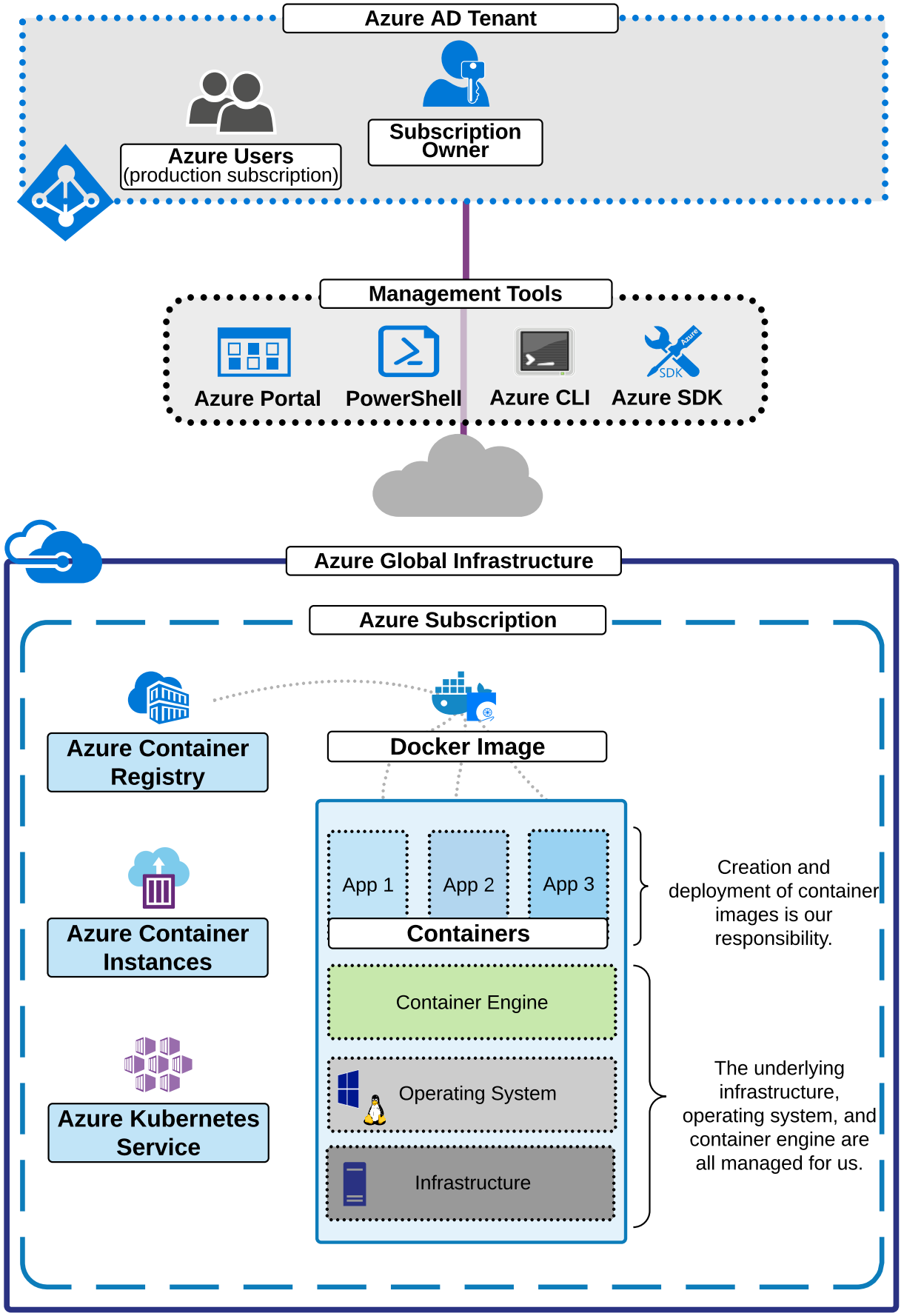
## Azure Containers

Azure provides a range of services that support containers.

These services span from a container image registry to a simple container engine, and finally to a fully orchestrated container cluster.

Go Back

Appendix



Subscription and Services Layer

Physical and Networking Layer

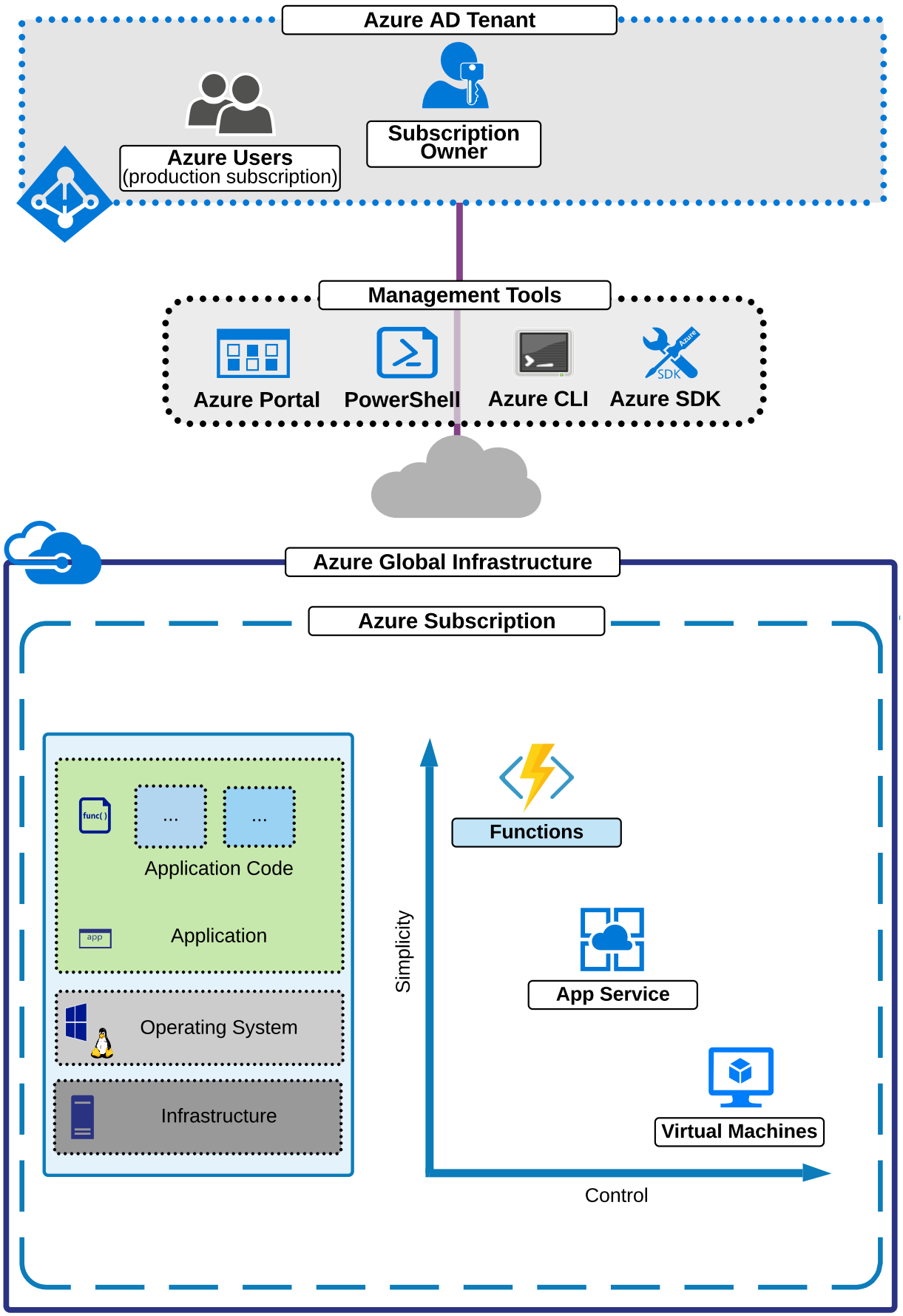
## Azure Functions

Azure Functions provides us the ability to run a small piece of code, which is focused on an individual task.

This diagram helps to illustrate the level of difficulty and control when we compare Functions to other servers.

Go Back

Appendix



Subscription and Services Layer

Physical and Networking Layer

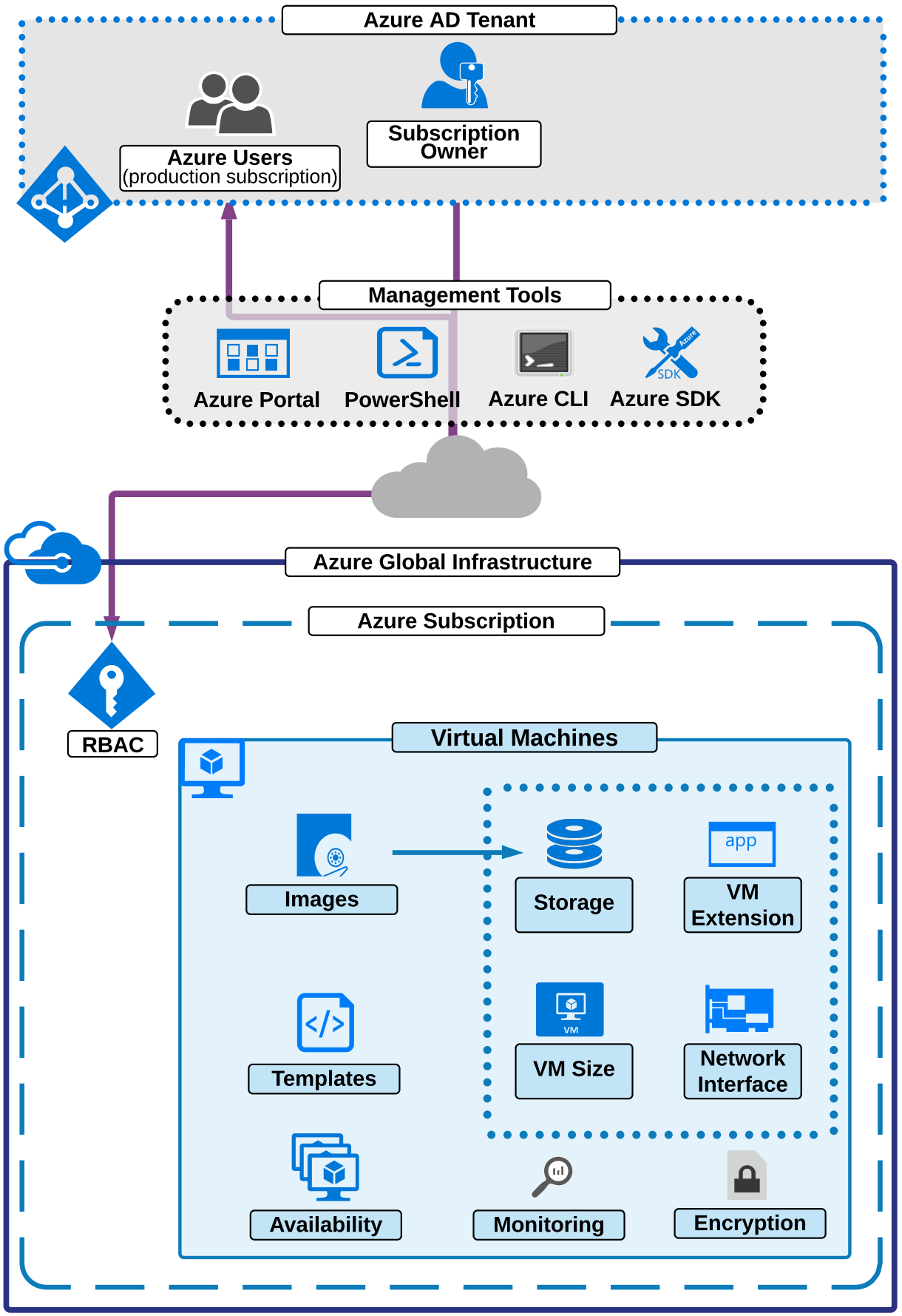
## Virtual Machines
(VMs)

This view provides a more in-depth overview of key Virtual Machine components.

See more about VM properties, as well as supporting functionality for high availability, monitoring, and automated deployments.

Go Back

Appendix



Subscription and Services Layer

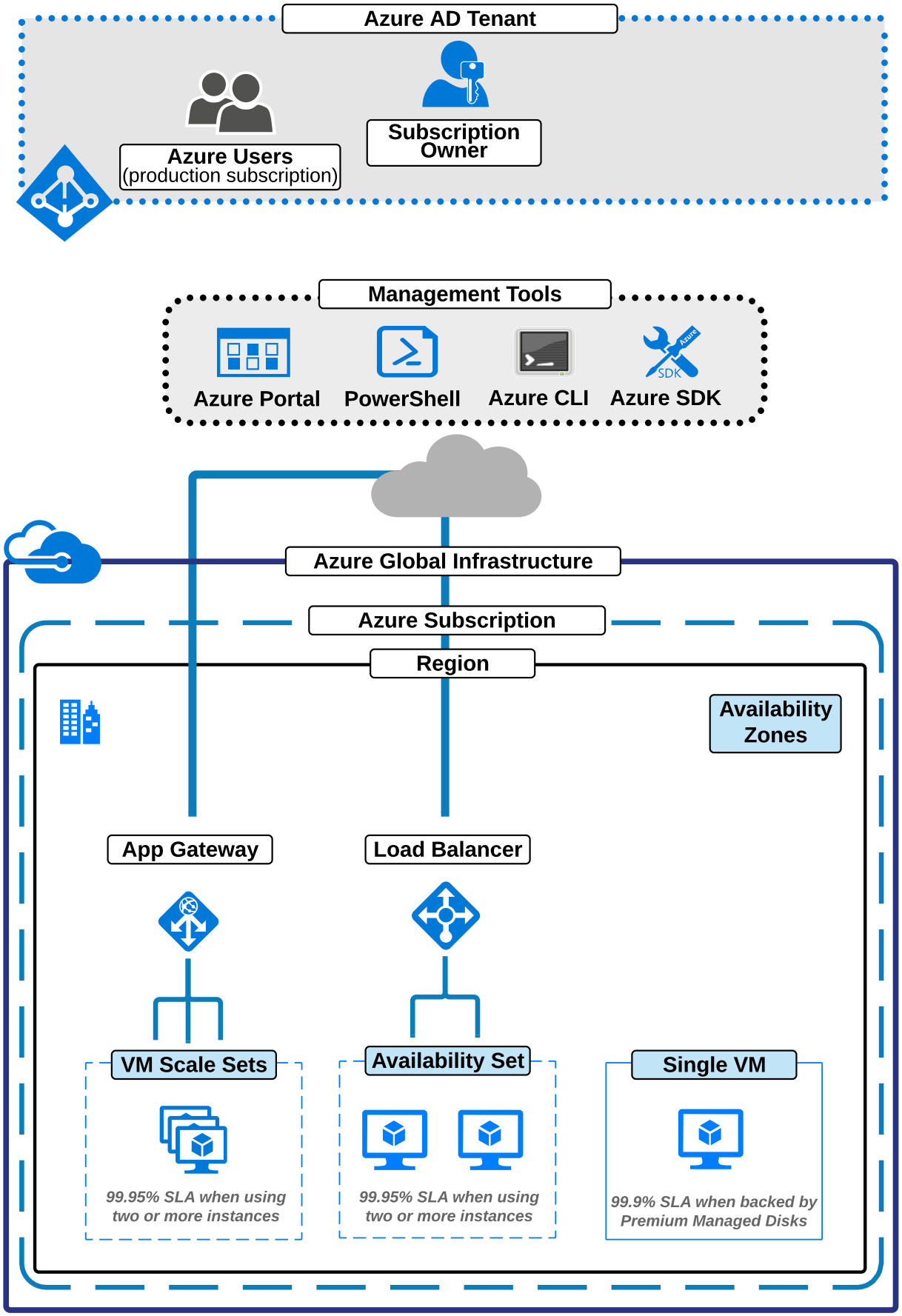
Physical and Networking Layer

## Virtual Machines
(High Availability)

Virtual Machines can be made highly-available through Availability Sets, and VM Scale Sets, in combination with other Azure services and architecture.

Microsoft also provides a Service Level Agreement (SLA) regarding the availability of VMs.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

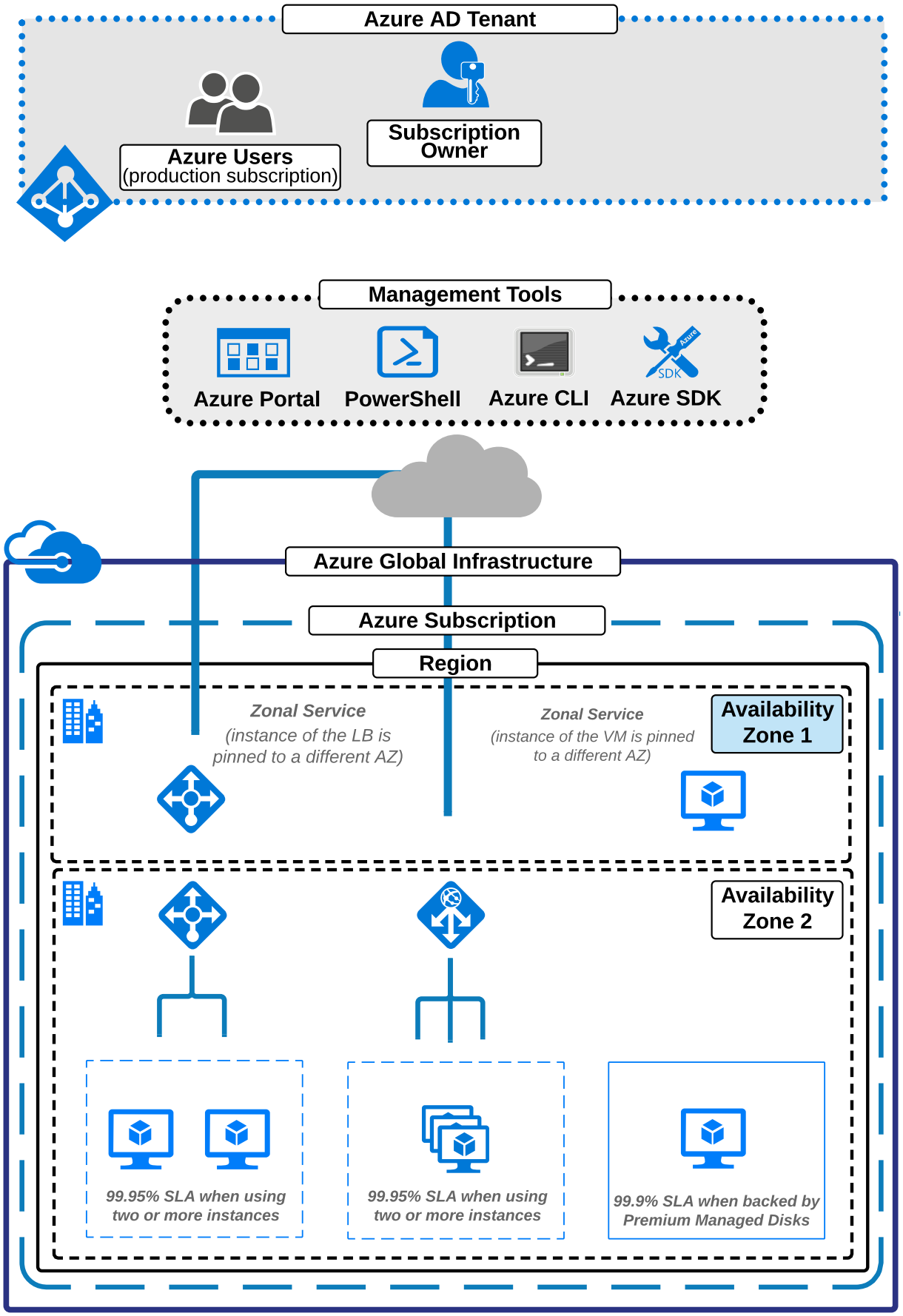
## Availability Zones

This page provides a high level view of Availability Zones when used for VMs.

Note that not all regions or services support Availability Zones; this is a newer solution underpinned by Microsoft's Global Infrastructure.

Go Back

Appendix



Subscription and Services Layer

Physical and Networking Layer

## Azure Functions

Description... Serverless...

Go Back

Appendix

Azure AD Tenant

Azure Users
(production subscription)

Subscription Owner

Management Tools

Azure Portal

PowerShell

Azure CLI

Azure SDK

Azure Global Infrastructure

Azure Subscription

Virtual Machines

Images

Templates

Managed Disks

VM Extension

VM Size

Network Interface

Availability

Monitoring

Subscription and Services Layer

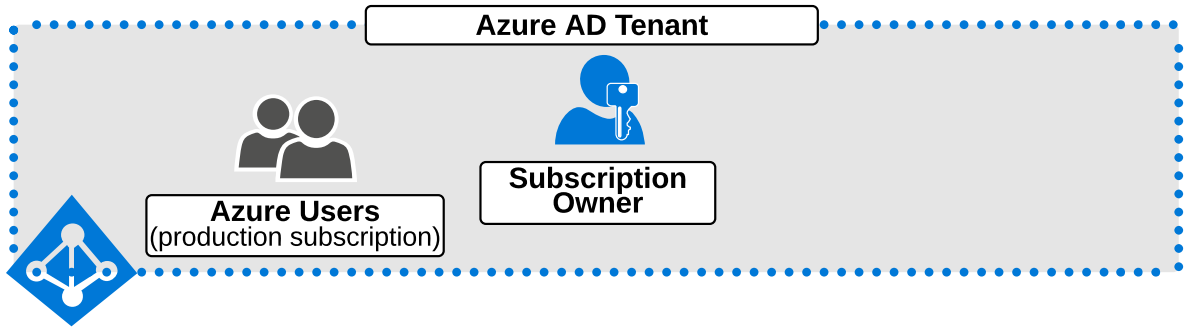
Physical and Networking Layer

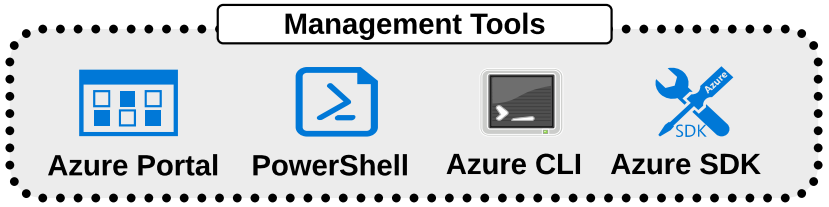
## Containers

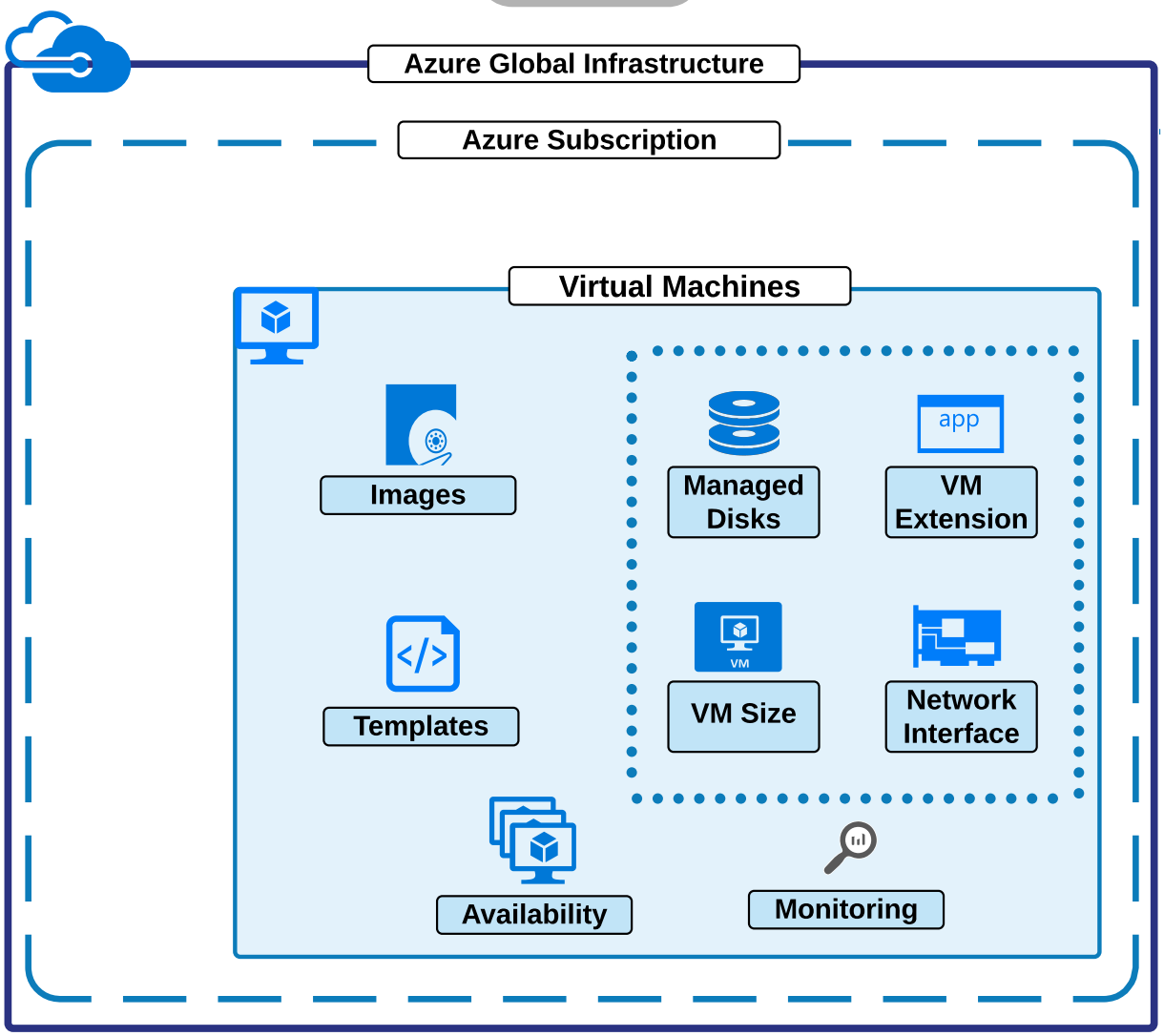
Description...

Go Back

Appendix



Close

# Azure Functions

Azure Functions enable rapid deployment of code which can be triggered by a range of events, all without the need of managing the underlying infrastructure. This capability is often referred to as "serverless."

Functions are created in a Function App, the parent container responsible for the underlying execution environment, settings, and other properties summarized below.

**Key features of Azure Functions:**

- Provide a simple way of rapidly deploying functional code without having to manage infrastructure
- Enable cost savings when using the consumption plan - pay only for execution time (detailed below)
- Integrate with a number of Azure services
- Support a range of programming languages, including C#, F#, JavaScript, and Python

**Key properties of Azure Function App / Functions:**

- Parent container is the Azure Function App, which defines the following properties:
  - Hosting Plan**:**
    - Consumption Plan - Manages underlying resources and scale; billed only for execution time
    - App Service Plan - Refer to the App Service popup for more information
  - Operating System (OS): Supports either Windows or Linux (which also supports containers)
  - Runtime Stack: The programming runtime stack for the environment (e.g. .Net); impacted by OS
- Functions are small code which operate in response to an event (trigger), and are typically made up of:
  - The actual code written in one of the supported programming languages
  - Triggers which cause the code to execute, including HTTP, timer, and a range of Azure services
  - Bindings which allow for connecting data with other services (both input and output)

Close

# Virtual Machine (VM) Images

VM Images help solve two main problems:
1. Performing an operating system (OS) installation within Azure is not possible.
2. Within the cloud, it's very beneficial to be able to pre-configure a VM and re-deploy it repeatedly.

For these reasons, we can either use readily available VM Images from the Azure Marketplace, create our own custom images from existing VMs, or upload a generalized VHD to blob storage.

**Information about Marketplace Images:**

- Can be deployed with ARM templates, PowerShell, or CLI, by specifying these image parameters:

| Parameter | Description | Example |
|---|---|---|
| **Publisher** | The organization which built and supplied the image | Canonical, MicrosoftWindowsServer |
| **Offer** | Typically represents a category/grouping of related images from a Publisher | UbuntuServer, WindowsServer |
| **SKU** | An instance of an offer | 16.04-LTS, 2016-Datacenter |
| **Version** | The version number of an image SKU | Determined by the Publisher. The term **latest** can be specified to use the latest version. |

**Information about Custom Images:**

- To create an image that can be used for deploying VMs, the OS of the VM must first be generalized.
- Generalizing removes unnecessary account and application specific information from an OS image.
- When you create an image from a VM, it can no longer be powered on or used as a VM normally.

| Operating System | How to Generalize |
|---|---|
| **Windows** | Generalize the VM (, OOBE, generalize); Deallocate the VM; Mark the VM as generalized; Create an image from the VM. |
| **Linux** | Generalize/Deprovision the VM ( ); Deallocate the VM; Mark the VM as generalized; Create an image from the VM. |

# Azure App Services

Close

Azure App Services is a platform-as-a-service (PaaS) solution where Microsoft manages the underlying compute resources and provides features and functionality to host web, mobile, and API apps.

The main components of Azure App Services are as follows:

- App Service: The actual instance of your mobile/web/api application, including all related settings
- App Service Plan: Container for the app service which defines features, underlying resources, and cost

**Below are some of the main features of Azure App Service:**

- Microsoft manages the underlying operating system, including updates
- Provides rapid deployment times, including support for GIT, FTP, and Web Deploy.
- ASP.NET, classic ASP, Node.js, PHP, and Python are supported.
- Azure allows processes to run in the background using WebJobs.
- Some limitations to be mindful of include:
  - There is no remote desktop access to App Service servers.
  - You cannot install custom applications or services.
  - Start-up tasks cannot be configured (though you can use WebJobs for scheduled tasks).
- It supports Windows, Linux, and containers.

**The following are some important facts about App Service Plans:**

- Generally speaking, there are three main types of App Service Plan:
  - **Dev/Test** - Limited resources and functionality including free/shared plans
  - **Production** - Dedicated higher tier resources with greater functionality and scalability
  - **Isolated** - Completely dedicated and isolated compute and network, with maximum scalability
- Plans determine the features available, such as scaling, SSL and custom domain support.
- Features and plans are constantly changing, for current information refer to these:
  - https://docs.microsoft.com/en-us/azure/app-service/overview-hosting-plans
  - https://azure.microsoft.com/en-au/pricing/details/app-service/linux/

**A Note about Function Apps:**

Function Apps can be deployed to an App Service Plan. But for a truer "serverless" experience, using a Consumption Plan is recommended instead.

Close

# Azure Container Registry (ACR)

Azure Container Registry (ACR) provides the ability to store images for your containerized applications. The underlying service is a docker registry based on Docker Registry 2.0, which is fully managed by Microsoft.

ACR also includes Azure Container Registry Build (ACR Build), a suite of tools which helps building container images and automating container development and maintenance processes.

**Key features of Azure Container Registry:**

- Private storage of docker-formatted images
- Supports tooling similar to Docker Registry 2.0
- Windows and Linux container images are supported within a single registry

**Configuration of Azure Container Registry:**

- Can be configured as either Basic, Standard or Premium pricing SKU
  - Differs in daily cost, included storage, total webhooks, and support for geo-replication (premium):
    - View current information at this Microsoft pricing article.
- Requires a registry name (accessible at *registryname.azurecr.io*)
- Supports authentication from Azure AD, managed identity, service principal, or an admin account

Close

# Azure Container Instances

Azure Container Instances (ACI) are one of the fastest options for deploying container applications within Azure, when compared to the other Azure container services.

ACI supports deployment of containers from both public repositories, like a Docker Hub or private repositories, when using Azure Container Registry.

**Important information about Azure Container Instances:**

- ACI is useful for isolated, simple applications. For multi-container apps, use Azure Kubernetes Service.
- Container groups (for Linux containers only) enable deployment of multiple containers to a single host.

**Configuring Azure Container Instances:**

- The image to create the container is required using a public or private repository.
- An ACI can be configured as either Linux or Windows, but note that Windows do not support Container Groups.
- ACI is publicly accessible using a unique DNS label *<dns-name-label>.<region>.azurecontainer.io*.
- It will soon be possible to deploy container instances to an Azure VNet. This feature is in preview.
- It is not possible to change the public IP address of container instances.

Close

# Azure Kubernetes Service (AKS)

Azure Kubernetes Service (AKS) is a Microsoft hosted and managed Kubernetes service that allows for easy deployment of a Kubernetes cluster to Azure.

Kubernetes is an open-source solution which helps manage multi-container environments. This includes scalability, automated deployments, and resource management for containerized applications. Kubernetes helps provide a focus on application workloads by abstracting the underlying infrastructure.

**Important information about AKS:**

- Deploying an AKS cluster results in two key components:
  - The Kubernetes cluster master:
    - This is managed by Microsoft, and does not incur any cost.
    - It includes kube-apiserver, etcd, kube-scheduler, and kube-controller-manager.
  - The Kubernetes nodes:
    - They are managed by you and *do* incur costs.
    - Microsoft recommends a minimum of 3 nodes for resiliency.
    - Node count can be changed, but node size cannot.
- To interact with your AKS cluster use:
  - The Azure portal and CLI
  - The command-line client
  - The Kubernetes dashboard (by using )

**Configuring AKS:**

- Creating an AKS cluster requires several details:
  - A name for your Kubernetes cluster
  - The version of Kubernetes you wish to deploy
  - A DNS name prefix for connecting to the Kubernetes API
  - VM size to be used for your cluster nodes (cannot be changed after creation)
  - The number of nodes for your cluster
  - An Azure AD Service Principal (used to create other Azure resources such as load balancers)
- AKS is deployed to two resource groups:
  - One which contains the Kubernetes service resource
  - Another (created by AKS) which contains all infrastructure associated with the cluster

Close

# Web Apps

Web Apps provides a managed hosting solution for deploying scalable web applications atop App Service.

**A summary of important Web App features is:**

- Functionality will be determined by the underlying App Service / App Service Plan
- Supports Windows and Linux platforms:
  - Windows supports: .Net, Java, Node.js, PHP and Python
  - Linux supports: .Net Core, Node.js, PHP and Ruby
- Can also be configured as Web Apps for containers
- Supports various developer and deployment features such as:
  - Continuous and integrated deployments
  - Support for Git, OneDrive, BitBucket, Azure Repos, etc.
- Provides the capability of integration with on-premises environments using Hybrid Connections

**Background tasks using WebJobs**

WebJobs are a feature of Web Apps which provide the ability to schedule either continuous or triggered tasks for running in the background of a Web App.

**A summary of important information about WebJobs is:**

- Continuous jobs:
  - Start immediately once created (and can be restarted if the job ends)
  - Run on all instances that the web app runs on, within the server farm (App Service Plan)
  - Store copies of runtime files in App_Data/jobs/continuous
- Triggered jobs:
  - Only start once triggered manually, or based on a schedule
  - Run only on a single instance within the App Service Plan as determined by Azure
  - Store copies of runtime files in App_Data/jobs/triggered
  - Support scheduling using CRON expressions (refer to this Microsoft article for more information)

# ARM Templates for VMs

Close

Azure Resource Manager (ARM) templates can be used to deploy *infrastructure as code*. This means using text files to represent infrastructure. For more information, refer to the Deployments page.

```
{
  "type": "Microsoft.Compute/virtualMachines",
  "name": "[variables('vmName')]",
  "location": "[resourceGroup().location]",
  "apiVersion": "2017-03-30",
  "dependsOn": [
    "[variables('storageRef')]",
    "[variables('nicRef')]"
  ],
  "properties": {
    "hardwareProfile": {
      "vmSize": "Standard_A2"
    },
    "osProfile": {
      "computerName": "[variables('vmName')]",
      "adminUsername": "[parameters('adminUsername')]",
      "adminPassword": "[parameters('adminPassword')]"
    },
    "storageProfile": {
      "imageReference": {
        "publisher": "MicrosoftWindowsServer",
        "offer": "WindowsServer",
        "sku": "2016-Datacenter",
        "version": "latest"
      },
      "osDisk": {
        "createOption": "FromImage"
      },
      "dataDisks": [
        {
          "diskSizeGB": 1023,
          "lun": 0,
          "createOption": "Empty"
        }
      ]
    },
    "networkProfile": {
      "networkInterfaces": [
        {
          "id": "[variables('nicRef')]"
        }
      ]
    },
    "diagnosticsProfile": {
      "bootDiagnostics": {
        "enabled": true,
        "storageUri": "[variables('storageRef')]"
      }
    }
  }
}
```

### Components of the VM Resource

It is important that you are familiar with the ARM Template, specifically for the VM resource. Below is a summary:

- **hardwareProfile:** VM size
- **osProfile**: Values required by the OS (hostname, credentials)
- **storageProfile**: The OS image to be used
- **osDisk**: Properties of the OS disk (caching, creation method)
- **dataDisks**: Array of properties for 1 or more data disks
- **networkProfile**: Network interface
- **diagnosticsProfile**: Monitoring and diagnostics

Close

# Virtual Machine Monitoring

Virtual Machine (VM) monitoring is based on the Azure Monitor service, and helps to provide visibility of VM metrics, diagnostic information, and logs. In addition to this, the Activity Log can be thought of as an audit-trail, and helps to provide visibility of actions that have been performed on the VM itself.

**Information about VM Monitoring:**

- Azure Monitor exposes host-level metrics (like CPU utilization, network usage, etc.) by default
- VM guest-level metrics, logs, and other diagnostic information require the Azure diagnostics agent:
  - Linux VMs require the "Linux Diagnostic Extension"
  - Windows VMs require the "Azure Diagnostics Extension"
  - The diagnostics agent is configured to store additional data in a storage account
  - A separate agent (Log Analytics agent) can be used to send data direct to a Log Analytics workspace

More information about monitoring and diagnostic can be found in the Monitoring section.

Close

# Availability Sets (AS)

Availability Sets (AS) are a tool for managing the high availability of Virtual Machines (VMs). In order to use an AS, you create the AS and then place VMs with a duplicate purpose in the AS.

The AS helps you to achieve the 99.95% Azure SLA by managing how VMs within the AS respond to planned (i.e. updates) or unplanned (i.e. faults) outages. This is managed through **update domains (UD)** and **fault domains (FD)** which are explained further below.

**Use cases:**
- Two more more duplicate file servers within an AS, ensuring one file server is always available to users
- Two or more duplicate web servers within an AS, ensuring one web server is always available to users

**Important information:**
- A single VM within an AS will not achieve the 99.95% Azure SLA; there must be at least two VMs present.
- Five (5) to twenty (20) update domains can be configured; the default value is five.
- Two (2) to three (3) fault domains can be configured, and this varies by region.

**Fault Domains (FD):**
Fault domains represent a common source of failure. For example, when the underlying Microsoft Azure infrastructure which the VMs operate on shares the same network switch, or power source.

**Update Domains (UD):**
Update domains are used to logically group VMs (and underlying hardware) which can be rebooted by Microsoft for the purpose of platform updates. *Only one UD will be rebooted at a time*.

| NAME | STATUS | FAULT DOMAIN | UPDATE DOMAIN |
|---|---|---|---|
| vm1 | Running | 0 | 0 |
| vm2 | Running | 1 | 1 |
| vm3 | Running | 0 | 2 |
| vm4 | Running | 1 | 3 |
| vm5 | Running | 0 | 4 |

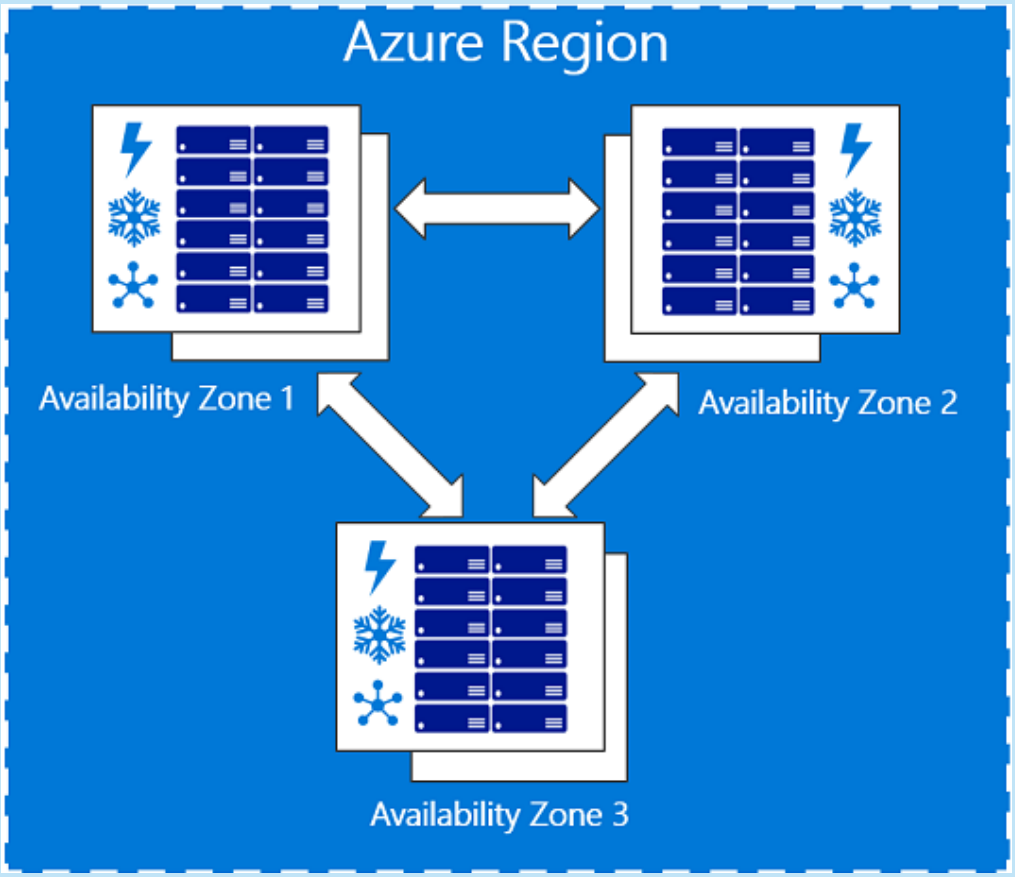
Close

# Availability Zones (AZ)

Availability Zones (AZ) are a relatively new service from Microsoft, and are focused on improving high availability. AZs are not yet available for all regions or services.

An AZ works at the Microsoft datacenter layer by providing isolated locations within an Azure region that have redundant power, cooling, and networking. There are two ways to use an AZ:

1. **Zonal services** - where you pin a resource to a specific zone (e.g. VMs, managed disks, load balancers)
2. **Zone-redundant services** - where the platform replicates across zones (e.g. ZRS storage, SQL)




**https://docs.microsoft.com/en-us/azure/availability-zones/az-overview**

# Virtual Machine Scale Sets (VMSS)

Close

A Virtual Machine Scale Set (VMSS) is essentially a service which provides the ability to dynamically scale Virtual Machines (VM) in and out based on demand.

There are two main components to a VMSS:
- The definition of the VM; image, size, connectivity, etc
- The autoscaling configuration; defining when, why, and how a VMSS will scale

**What is an Autoscale setting:**
Enabling autoscaling for VMSS requires an Azure Monitor Autoscale setting. An Autoscale setting is comprised of the following:
- One or more profiles - profiles define default, maximum, and minimum capacity, as well as any rules
  - Regular/default profile - default profile which includes rules to scale based on metrics
  - Fixed date profile - profile which applies only on a certain single date/time (e.g. January 1, 2019)
  - Recurrence profile - profile which applies on recurring date/time ranges (e.g. Monday - Friday)
- One or more metric rules - used to define what a criteria and action
  - Condition - specific metric criteria to occur, e.g. average CPU reaches 70%
  - Response - specific action to apply if the criteria occurs, e.g. increase instances by 1
- Note that metric rules are optional for fixed/recurring profiles

**Which profile will Autoscale run?**
- Fixed date profiles will run first; if there are multiple fixed date profiles, Autoscale will select the first one
- If no fixed date profiles exist, Autoscale will run any recurrence profiles that exist
- If there are no fixed date or recurrence profiles, Autoscale will run the regular/default profile

**Which rules will Autoscale run?**
- Autoscale will select which rules to run, after it determines which profile should run
- Scale-out rules (direction = increase) will run first
- If there are multiple scale-out rules, Autoscale will use the largest capacity
- If no scale-out rules should trigger, Autoscale will evaluate scale-in rules (direction = decrease)
- If there are multiple scale-in rules, Autoscale will use the largest capacity

Close

# Virtual Machines (VM)

Virtual Machines (VMs) can be thought of as virtual computers which operate on the Azure platform. Using an image, the VM can run an Operating System as if it were a real physical server.

VMs are considered Infrastructure as a Service (IaaS), which at a very high level means that you manage most of the environment, except for the underlying physical infrastructure.

**Use cases:**

- VMs can be used for operating servers in the cloud, similar to how you would currently do something, like running an on-premises file or a web server.

**Key information about Virtual Machines:**

- VMs have important characteristics that define how they behave, such as size, image, network, etc.
- Both Windows and Linux VMs are supported in Azure, but only the 64-bit versions.

**Key information about Virtual Machine SLA:**

- Microsoft will only provide an SLA for a Virtual Machine under the following conditions:
  - When running a single VM only, with premium storage being required for all OS disks and data disks
  - When running two or more instances of a VM within the same Availability Set
  - When running two or more instances of a VM across two or more Availability Zones in the same region

More information can be found at the Microsoft article, here:
https://azure.microsoft.com/en-us/support/legal/sla/virtual-machines/v1_8/

# Virtual Machine (VM) Storage

Close

Overview | Encryption

Virtual Machines disks are housed in Azure storage. A VM disk is where the operating system (OS) is installed, where applications are installed, and where data resides.

There are three main disks used by VMs:
1. **Operating system disk**: based upon a VM image which includes the necessary operating system files
2. **Data disks**: can be used as desired and are typically created as a blank disk
3. **Temporary disk**: included with all VMs by default for temporary purposes only - data can be lost

**Important information about VM disks:**
- All VM disks are essentially virtual hard disks (VHDs) stored in an Azure storage account as page blobs
- VHD files can be used as a source to create disks or images (e.g. if you upload a VHD)
- VM disks can either be managed or unmanaged:
  - Unmanaged disks require manual configuration of the storage account (SA) the disk resides in.
  - Managed disks are still housed within an SA, however this is managed by Microsoft to provide greater simplicity, reliability, and control over performance.
  - Managed disks are independent resources, whereas unmanaged disks belong to the VM resource.
- A VM will l  always include an operating system disk, and a temporary disk, at a minimum
- VM disks can be one of three performance tiers:
  - Standard HDD disks - cost effective, low IOPS, suitable for bulk storage, backups, etc.
  - Standard SSD disks - similar to Standard HDD disks, but greater consistency and reliability.
  - Premium SSD disks - high performance, low latency disks, accessible to VMs with an 's' in the name.
- For high-performance applications that require high IOPS, Premium SSD Disks should be used
- See more information here:
https://docs.microsoft.com/en-au/azure/virtual-machines/linux/about-disks-and-vhds

**Important information about securing data:**
- Azure Storage Service Encryption safeguards VM disks at the Azure infrastructure/data level.
- Azure Disk Encryption also protects at the VM disk resource level, protecting OS/boot and data disks.
- Refer to the Security page for more information.

# Azure Disk Encryption

Close

Overview | Encryption

Azure Disk Encryption (ADE) encrypts data for either Windows or Linux VM disks. It helps protect against the risk of data theft from a disk that an unauthorized individual may gain access to.

**Important information about Azure Disk Encryption:**

- Windows VM disks are protected with BitLocker.
- Linux VM disks are protected with DM-Crypt.
- Azure Disk Encryption provides volume encryption for the OS and data disks.

**Enabling Azure Disk Encryption:**

- ADE can be enabled in a number of ways, including PowerShell, CLI, and templates.
- The following commands can be used:
  - PowerShell: `Set-AzureRmVMDiskEncryptionExtension`
  - CLI: `az vm encryption enable`
- Be aware that the latest release of Azure Disk Encryption relies on a Key Vault.
- The Key Vault used for ADE must have 'enable access to ADE for volume encryption' selected.

# Virtual Machine (VM) Sizes

Close

The size of a virtual machine (VM) specifies important characteristics of the virtual hardware available to a VM. This is not just CPU and memory, but also network, storage, and other key properties.

It is important to be familiar with these characteristics and the various VM families, but the AZ-300 does not require you to memorize specific resource amounts or figures.

**Important information about VM sizes:**

- VM size defines the vCPU and memory for a VM.
- The maximum number of network interfaces, or disks, that can be attached is limited by the VM size.
- Maximum disk IOPS and network bandwidth is also limited by the VM size.

**Below is a summary of current VM families:**

See https://docs.microsoft.com/en-us/azure/virtual-machines/windows/sizes for updated details.

| Type | Sizes | Description |
|---|---|---|
| General purpose | B, Dsv3, Dv3, DSv2, Dv2, Av2, DC | Balanced CPU-to-memory ratio. Ideal for testing and development, small to medium databases, and low to medium traffic web servers. |
| Compute optimized | Fsv2, Fs, F | High CPU-to-memory ratio. Good for medium traffic web servers, network appliances, batch processes, and application servers. |
| Memory optimized | Esv3, Ev3, M, GS, G, DSv2, Dv2 | High memory-to-CPU ratio. Great for relational database servers, medium to large caches, and in-memory analytics. |
| Storage optimized | Lsv2, Ls | High disk throughput and IO ideal for Big Data, SQL, NoSQL databases, data warehousing and large transactional databases. |
| GPU | NV, NVv2, NC, NCv2, NCv3, ND, Ndv2 (Preview) | Specialized virtual machines targeted for heavy graphic rendering and video editing, as well as model training and inferencing (ND) with deep learning. Available with single or multiple GPUs. |
| High performance compute | H | Our fastest and most powerful CPU virtual machines with optional high-throughput network interfaces (RDMA). |

# Virtual Machine (VM) Extensions

Close

Virtual Machine (VM) extensions are generally lightweight applications or services which assist with post-deployment configuration, or various automation tasks on VMs.

Extensions can be enabled through the Portal, CLI, PowerShell, and with ARM Templates.

**Use cases:**

- Using the Custom Script Extension to change the timezone and language settings for a newly deployed VM
- Using the Azure Diagnostics Extension to support additional monitoring of VMs

| NAME | TYPE | VERSION | STATUS |
|---|---|---|---|
| Microsoft.Insights.VMDiagnosticsSettings | Microsoft.Azure.Diagnostics.IaaSDiagnostics | 1.* | Provisioning succeeded |

| NAME | TYPE | VERSION | STATUS |
|---|---|---|---|
| LinuxDiagnostic | Microsoft.Azure.Diagnostics.LinuxDiagnostic | 3.* | Provisioning succeeded |

Close

# Network Interfaces

Network Interfaces (also known as Network Interface Cards, or NICs) are independent resources which, when connected to a Virtual Machine (VM), enable network connectivity with other networked resources.

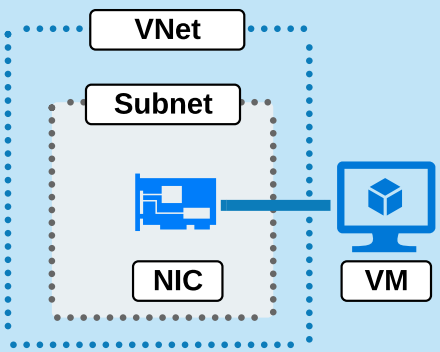
**Use cases:**

- VM network communication with the Internet, Azure, and even on-premises (via VPN or ExpressRoute)

**Important information about VM NICs:**

- NICs are independent resources which are created separately from a Virtual Machine
- When a VM is configured to forward packets (e.g. as a firewall), *IP Forwarding* must be enabled for the NIC
- It is recommended to not statically assign private IP addressing within the operating system (OS) of a VM
- If the VM is using a NIC with multiple IP addresses, then static configuration *will* be required within the OS
- DNS settings can be specified manually for an individual NIC, or be inherited from a VNet

As the NIC is ultimately a network resource, greater detail can be found on the Networking page here.



## Coming Soon

Close

The AZ-300 course is released **in preview**, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

To continue, please click the close (X) button.

Subscription and Services Layer

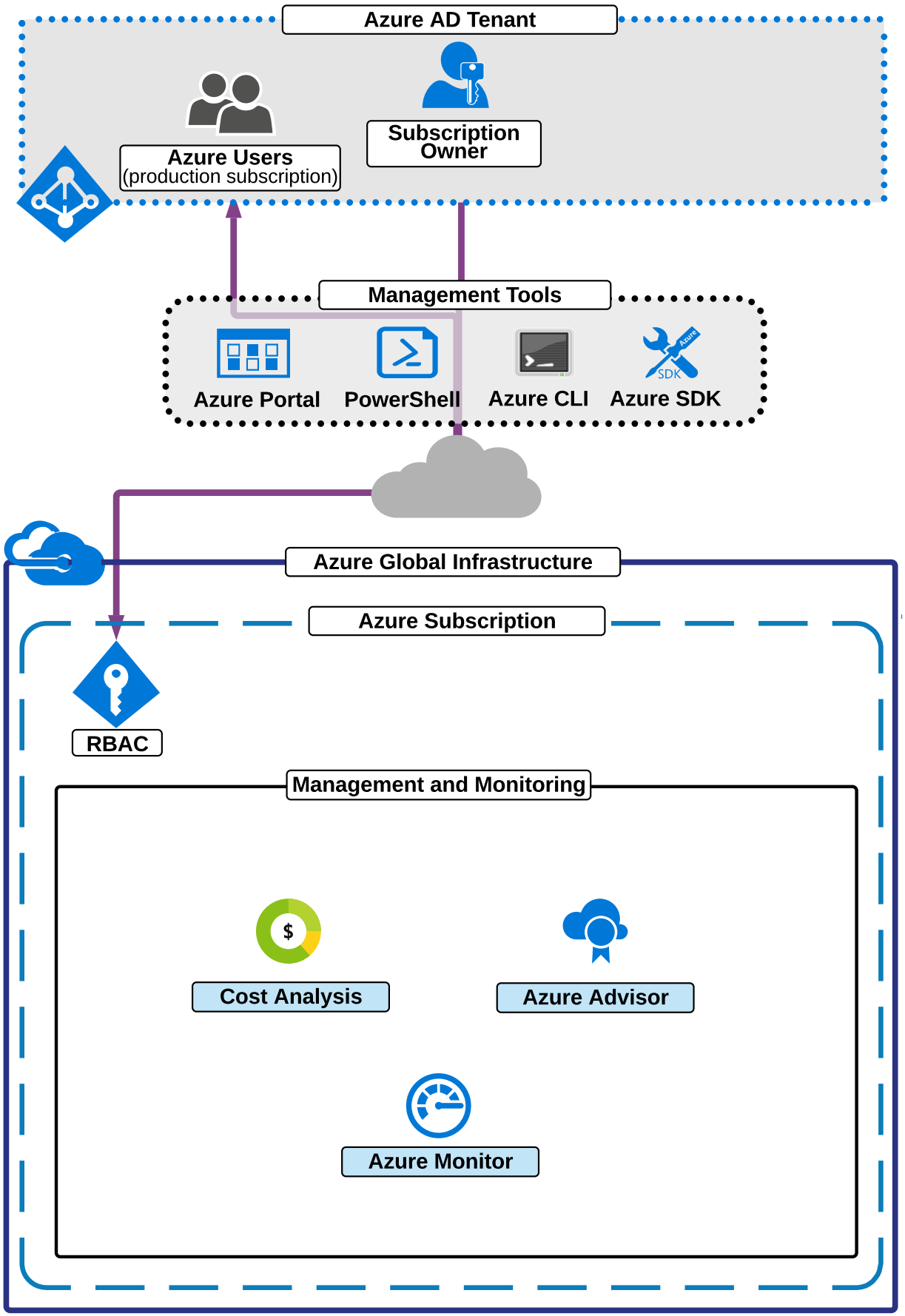
Physical and Networking Layer

## Monitoring and Management

Microsoft provides a range of tools to manage and monitor your environment.

In this page you can find a link to the main services you must be familiar with for the AZ-300 exam.

Appendix



Subscription and Services Layer

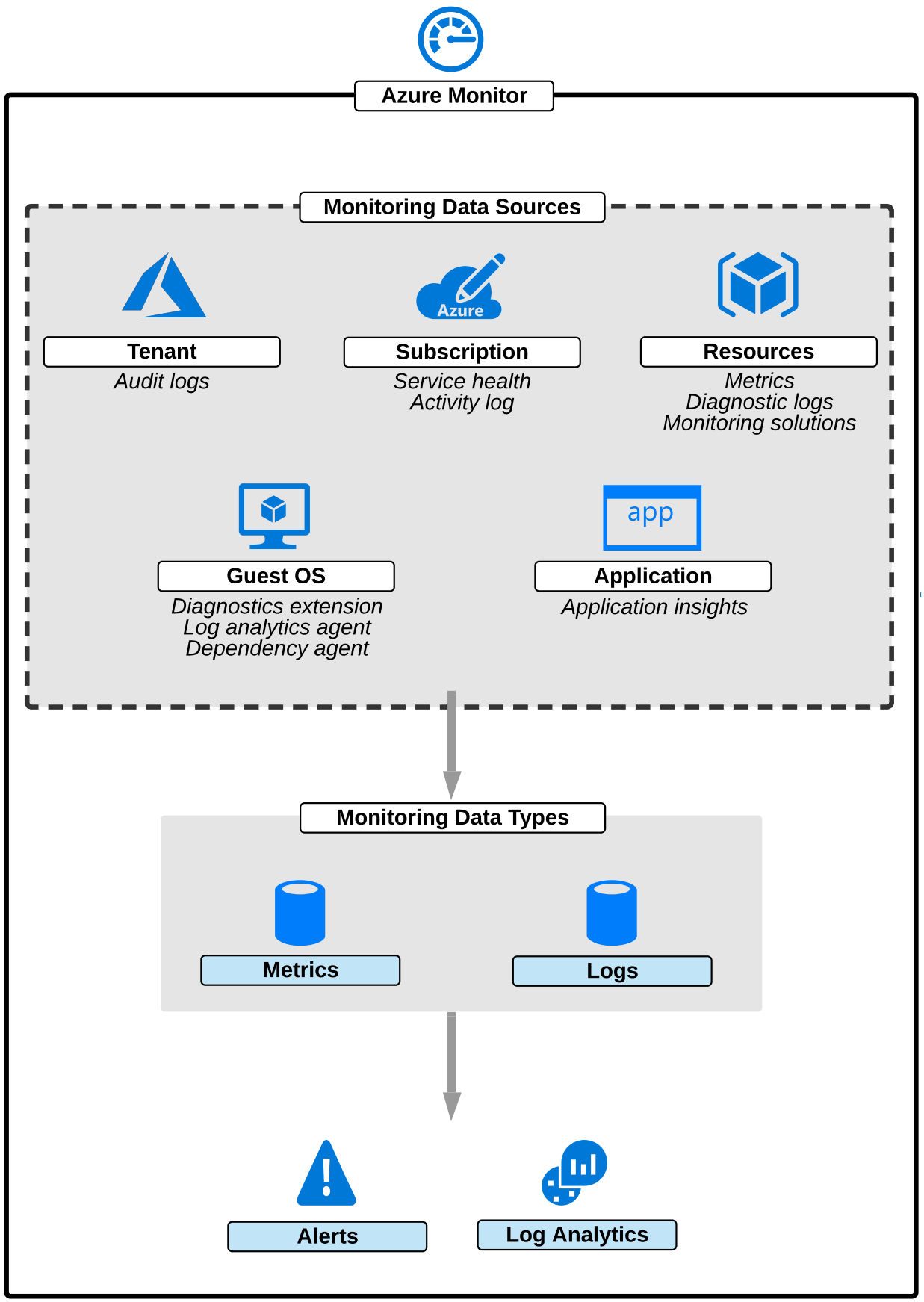
Physical and Networking Layer

## Azure Monitor

Azure Monitor is a collection of services which help monitor and diagnose Azure resources.

This page provides an overview of Azure Monitor; where data is collected from, the two fundamental types of data, and finally some of the main functions within the Azure Monitor toolkit.

Go Back Appendix



# Logs - Overview



Logs are larger sets of textual data, containing detailed descriptions and information about a specific resource, application, or activity. Data is typically collected sporadically.

**Examples of logs:**

- Activity logs from Azure resources
- Detailed diagnostics collected from Windows and Linux VM guest operating systems and applications
- Resource-level diagnostic logs provided through the Azure platform
- Custom log data

https://docs.microsoft.com/en-us/azure/azure-monitor/platform/tutorial-dashboards

More information can be found at the following Microsoft article:
https://docs.microsoft.com/en-us/azure/azure-monitor/platform/data-collection#logs

# Cost Management

Close

Overview | Cost Analysis | Azure Advisor

Cost management is an important part of any Azure Solution Architect's job. These are a number of tools we can utilize to help manage costs:

- Costs can be monitored and reported on through the Azure Portal, within the given subscription.
- Tagging resources helps to map costs according to categories we decide (e.g. department or application).
- Azure Advisor provides recommendations which help to optimize spend across Azure.

To help estimate costs of services ***before*** they have been provisioned, you can use the Azure Pricing Calculator at the following link: https://azure.microsoft.com/Pricing/Calculator.

# Cost Analysis



Microsoft provides a number of features within the Azure Portal to help manage costs. It is possible to view current costs, see cost projections, analyze current spend, and view invoicing and billing information.

**Tips for analyzing and managing costs:**

- Show costs based on resource tags: *Subscription > Cost Management > Cost Analysis*
- See previous invoices: *Subscription > Billing > Invoices*
- View costs by resource: *Subscription > Overview*
- Look at a cost forecast: *Subscription > Overview*

To provide access to billing information, it is recommended that the Billing Reader role is used. More information can be found here: https://docs.microsoft.com/en-us/azure/billing/billing-manage-access.

# Azure Advisor



Azure Advisor is a free service provided by Microsoft which provides personalized recommendations tailored to your environment. Azure Advisor focuses on high availability, performance, security, and cost.

**Azure Advisor provides cost recommendations to help:**

- Optimize virtual machine spend by resizing or shutting down underutilized instances
- Reduce costs by eliminating unprovisioned ExpressRoute circuits
- Buy reserved virtual machine instances to save money, when compared to pay-as-you-go costs
- Reduce costs by deleting or reconfiguring idle virtual network gateways

More information on the other recommendations provided by Azure Advisor can be found here: https://docs.microsoft.com/en-us/azure/advisor/advisor-overview.

# Logs - Diagnostic Logs

Close



Diagnostic logs is typically rich and frequent data that details the operation of a given service or application. Exactly what they look like differs on a case-by-case basis.

**There are three types of diagnostic logs:**

- Tenant logs: Logs from outside of an Azure subscription (e.g. Active Directory logs)
- Resource logs: Logs emitted by resources deployed within an Azure subscription (e.g. storage accounts)
- OS-level logs: Logs collected by an agent running inside a compute resource (e.g. virtual machine)

**Important information about diagnostic logs:**

- Azure Monitor is able to provide diagnostic logs for a number of resources, with minimal configuration.
  - Logging for these resources can be enabled within Azure Monitor, in *Diagnostic Settings*.
  - These logs are collected by the Azure platform itself.
  - More information on supported resources can be found here: https://docs.microsoft.com/en-us/azure/azure-monitor/platform/tutorial-dashboards
- OS-level logs require an additional agent and configuration, in order to capture the log data
  - For virtual machines, this agent is installed using a VM extension

**What happens to the logs?**

- Azure Monitor uses *diagnostic settings* to configure three possible destinations:
  - A storage account, for archival
  - Log Analytics, for search and analytics
  - Event Hubs, to be integrated with other solutions
- When using the OS-level Azure Diagnostics Extension, data is stored in a specified storage account

# Logs - Activity Logs

Close

Overview | Diagnostic Logs | Activity Logs

Activity Logs, previously known as *Audit Logs* or *Operational Logs*, provide information about events that have occurred across a subscription. These logs are used to determine "who, what, and when" for operations performed on resources within a subscription.

**Important information about Activity Logs:**

- Subscription level log for write operations (PUT, POST, DELETE)
- Does not include read operations (GET)
- Does not include operations for resources that use the classic/"RDFE" model

There are a range of sources for Activity Logs:

- Administrative: Create, update, and/or delete actions performed through Azure Resource Manager
- Service Health: Service health incidents that have occurred in Azure
- Resource Health: Resource health events for resources (available, unavailable, degraded, unknown)
- Alert: A record of any Azure alerts activating
- Autoscale: Events relating to the operation of any autoscale settings you have defined
- Security: Alerts generated by Azure Security Center
- Policy: Reserved for future use by Microsoft

More information can be found here:
https://docs.microsoft.com/en-us/azure/azure-monitor/platform/activity-logs-overview.

# Metrics

Close

Metrics are lightweight numerical values that describe something about a resource over a given time period.

A number of metrics are provided by the Azure platform and are available by default, similar to log data. Additional metrics are also available from other sources, like application metrics via the Application Insights instrumentation, or OS-specific metrics via the diagnostics VM extension.

**Examples of metrics:**

- Average CPU utilization (as a percentage over time)
- Network throughput (kilobytes per second)

**Important information about metrics:**

- Metrics are collected at a frequency of one-minute by default and stored for 93 days
- There are many metrics available by default with various Azure resources
- Some additional metrics require further configuration (e.g. VM Diagnostics Extension) before they are available
- Metrics can be viewed through Azure Monitor, and also within the resource tab itself

More information can be found at this Microsoft article:
https://docs.microsoft.com/en-us/azure/azure-monitor/platform/data-collection#metrics

# Alerts - Overview

Close

Overview | Action Groups

Alerts are a feature of Azure Monitor which provide the ability to *perform some action(s)* in response to *something happening*.

In order to configure alerts, we must create an **alert rule** (including a name and description), which defines an **action group** to trigger, based on a **condition** being met for a given **resource**.

**Important information about alerts:**

- Alert conditions can only be based on two signal types: metrics, or activity logs.
- It's important to understand limits associated with the actions we can perform
  - Refer to the Action Groups tab above
- Triggered alerts are visible within *Azure Monitor > Alerts*, where they can be managed
  - Alert state management allows us to use three state alerts: new, acknowledged, and closed
  - Alert states are independent of the underlying monitoring condition
  - Historical information about the alert states, and monitor condition, are recorded

Close

# Alerts - Action Groups



Action groups are used within alerts to define *what should happen* when an alert is triggered. This can include emailing users, calling an Azure Function, and much more.

Below is a list of the types of actions that can be performed, as well as the limits per action group.

**Action groups (limits per action group displayed in parenthesis):**

- Email (up to 1000): Email notifications to an email address
- SMS (up to 10): SMS notifications to a phone number
- Push (up to 10): Push notification to the Azure app
- Voice (up to 10): Voice calls to a phone number
- Azure Function: Call to a Function app
- LogicApp (up to 10): Call to a Logic app
- Webhook (up to 10): URI call to a webhook
- ITSM (up to 10): Call to supported ITSM via an ITSM Connection
- Automation Runbook (up to 10): Call to a built-in or user Runbook

**Rate limiting for certain notification types:**
Microsoft imposes certain rate limits to ensure that alerts are manageable. To achieve this, notifications may be suspended if too many are sent to a particular phone number, email address, or device.

The rate limit thresholds are:

- 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

Refer to the following Microsoft article for more information:
https://docs.microsoft.com/en-us/azure/azure-monitor/platform/alerts-rate-limiting

# Log Analytics - Overview

Close

Overview | Log Search

Log Analytics is a service within Azure Monitor which helps to centralize log management for a variety of Azure services, resources, and other log data.

Log Analytics provides the storage and search functionality of logs within Azure Monitor. To send data to Log Analytics, **Diagnostic Settings** must be configured for a given resource.

**Important information about Log Analytics:**

- Log Analytics provides uses a powerful query language, based on the Data Explorer query language.
- All log data collected by Azure Monitor is stored within a Log Analytics workspace:
  - A workspace must be created for log data to be stored within Log Analytics.
  - All Log Analytics queries are scoped to the respective workspace by default.
  - Workspace data sources can include Azure resources, storage accounts, activity logs, VMs, etc.
- Historically, Log Analytics was a standalone service, however it is now part of Azure Monitor

**Important information about Log Analytics data sources:**

- Data sources can include Azure resources, activity logs, storage accounts logs, and virtual machines
- Diagnostic Settings are used to configure resources to send data to a Log Analytics workspace

# Log Analytics - Log Search

Close

Overview | Log Search

Log Analytics queries provide a way to perform powerful analysis of large volumes of log data. Microsoft manages the underlying analysis infrastructure so you do not have to (based on the Data Explorer service).

Below are some tips for log queries:

- Log Analytics data is organized in tables, which can be queried by piping the table through to an operator
- The below query shows a maximum of 50 results from the SecurityEvent table

```
SecurityEvent | limit 50
```

- The below query searches the Azure Activity Log for evidence of Storage Account keys being changed

```
AzureActivity | where OperationName == "Regenerate Storage Account Keys"
```

**Queries can be saved as functions to simplify advanced queries**

- For example, if the preceding query string is saved as "sakeyregen", then we could re-use it as follows:

```
sakeyregen | summarize count(Caller) by Caller
```

- This would take the output of the earlier command, and pipe it to the second command.
- The result of this query is to list and count all distinct users who have regenerated a storage account key.

**More information:**
The log query language is very flexible and powerful. Consequently, it can either be simple or complex. For this reason it is recommended you are familiar with the following article:

https://docs.microsoft.com/en-us/azure/azure-monitor/log-query/get-started-queries

Subscription and Services Layer

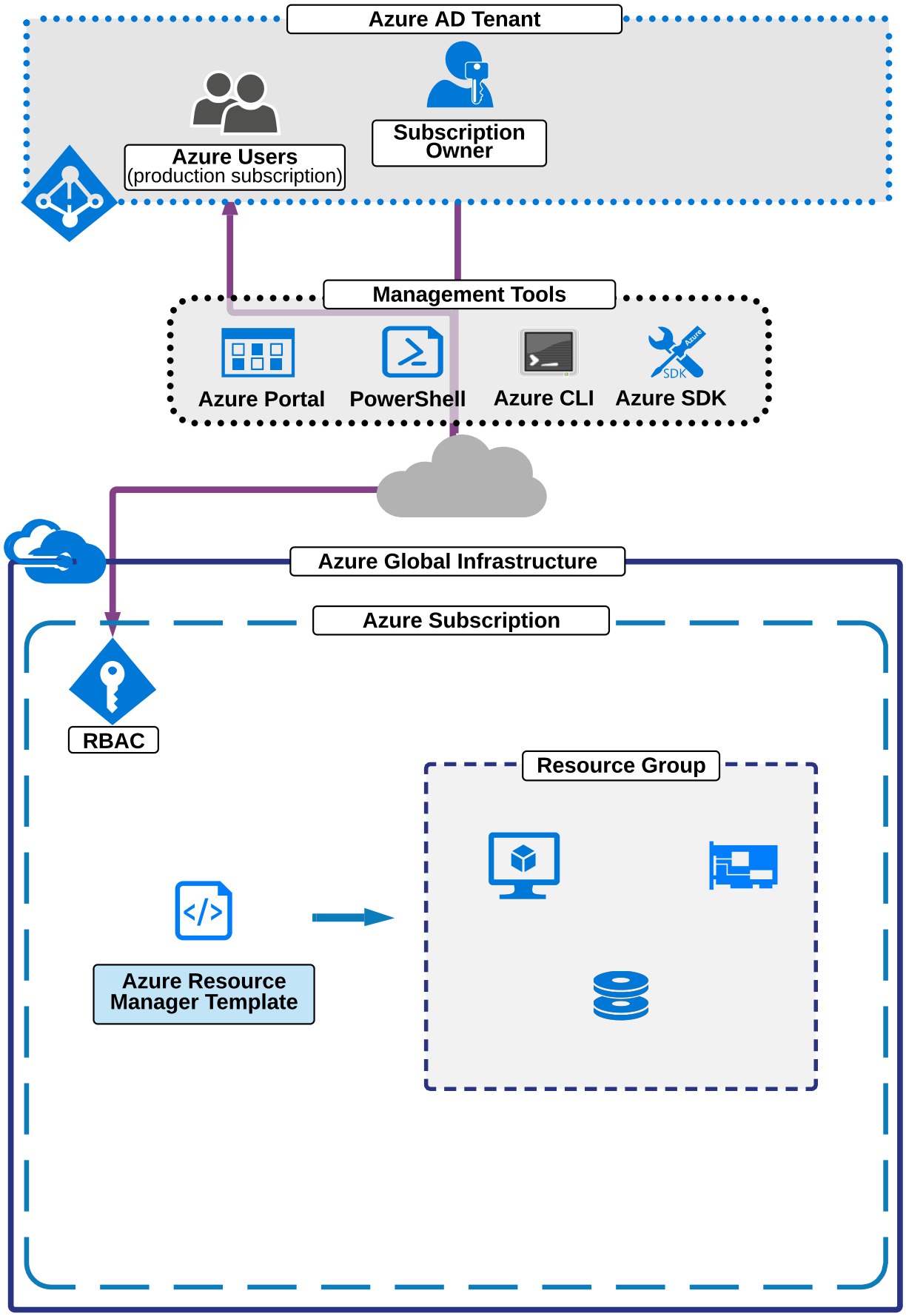
Physical and Networking Layer

## Automated Deployments

At the core, of automated deployments within Azure, is the Azure Resource Manager (ARM) Template. This template allows infrastructure to be defined as code.

Using ARM Templates, we can automate the deployment of infrastructure with scripts or code.

Appendix



## Coming Soon

Close

The AZ-300 course is released **in preview**, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

To continue, please click the close (X) button.

# Azure Resource Manager (ARM) Templates



Azure Resource Manager (ARM) Templates essentially allow us to use text to represent infrastructure. That is why this form of deployment is often called infrastructure-as-code.

**General Information on ARM Templates**

- ARM Templates use **declarative JSON syntax** to represent the infrastructure as code/text.
- We can automate the deployment of infrastructure using these templates with scripts or code.
- ARM Templates have a special structure (detailed more within the *Structure* tab above) allowing us to define multiple resources which are ***deployed to a Resource Group***.

**Information about saving ARM Templates**

- You can create and save your own ARM Templates to JSON files, and deploy them as illustrated below.
- Existing resources can be exported to ARM Templates via the portal:
  - Original template: Navigate to a Resource group, then **Deployments**, where you can view details/files for past deployments
  - Generated template: Navigate to a Resource, then **Automation script**, to view what was automatically generated

**Information about deployment modes**

- Incremental and Complete are the two modes that deal with *already existing* Resources in a Resource Group that you deploy an ARM Template to.
- Whichever mode is used, resources defined in the ARM Template, resources that already exist, will have their properties updated to match the ARM Template.
  - **Incremental mode:** Existing resources will be left unchanged if they are not specified in the template.
  - **Complete mode:** Resources in the Resource group but not the template *will be deleted*.

**Scripted deployment (using Azure CLI and PowerShell):**

```
az group deployment create \
  --name deploymentTest \
  --resource-group lab01rg \
  --template-file vmdeploy.json \
  --parameters vmName=testvm01

New-AzureRmResourceGroupDeployment -Name deploymentTest `
  -ResourceGroupName lab01rg `
  -TemplateFile c:\lab\vmdeploy.json `
  -TemplateParameterFile c:\lab\vmdeploy.parameters.json
```

# ARM Template Structure

Close

Overview | Structure | Example

ARM Templates are very versatile, so template files can either be large and complex, small and simple, or anywhere in between. At the very core, all ARM Templates are made up of the following elements:

- **$schema and contentVersion**: Required details describing the version and format of the template:
  - These values rarely change and are primarily used by Microsoft to describe the template structure.
- **Parameters**: Custom values you can use at deployment time to change what is being deployed:
  - Parameters help simplify your template by allowing you to have a single template with values which may change from time to time, rather than creating multiple templates with fixed/unchangeable values.
    - For example, you might have VMNAME or OPERATINGSYSTEM as parameters.
- **Variables**: Values used within the template, largely to help simplify the template usability and readability:
  - Variables can help having to re-write (or even re-calculate) the same piece of information repeatedly.
    - For example, a variable might include a reference to a subnet, in which you will deploy multiple VMs.
- **Functions**: User-defined functions which you can create and call within a template
- **Resources**: The very specification of the resource(s) you wish to deploy with the template
- **Outputs**: Values you wish to be returned after deployment

Note that , , and are always required within an ARM Template. The other elements do not have to be used.

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {…
  },
  "variables": {…
  },
  "functions": [
  ],
  "resources": […
  ],
  "outputs": {
  }
}
```

# ARM Template Example

Close

Overview | Structure | Example

The following is a breakdown of a Virtual Machine resource defined within an ARM Template. This is an excerpt only, and helps to explain the elements that make up a resource, within an ARM Template.

```
{
  "apiVersion": "2017-03-30",
  "type": "Microsoft.Compute/virtualMachines",
  "name": "[parameters('virtualMachineName')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
  ],
  "properties": {
    "hardwareProfile": {
      "vmSize": "[parameters('virtualMachineSize')]"
    },
    "osProfile": {
      "computerName": "[parameters('virtualMachineName')]",
      "adminUsername": "[parameters('adminUsername')]",
      "adminPassword": "[parameters('adminPassword')]"
    },
    "storageProfile": {
      "imageReference": {
        "publisher": "[variables('imagePublisher')]",
        "offer": "[variables('imageOffer')]",
        "sku": "[parameters('windowsOSVersion')]",
        "version": "latest"
      },
      "osDisk": {
        "createOption": "FromImage"
      },
      "dataDisks": [
        {
            "lun": 0,
            "name": "[concat(parameters('virtualMachineName'),'datamdd1')]",
            "createOption": "Empty",
            "caching": "None",
            "diskSizeGB": "[parameters('diskSizeGB')]",
            "managedDisk": {
              "storageAccountType": "[parameters('diskType')]"
            }
        }
      ]
    },
    "networkProfile": {
      "networkInterfaces": [
        {
          "id": "[resourceId('Microsoft.Network/networkInterfaces',variables('nicName'))]"
        }
      ]
    },
    "diagnosticsProfile": {
      "bootDiagnostics": {
        "enabled": false
      }
    }
  }
```

Below is a brief description of the properties and objects included within the VM definition:

- type: We are creating a VM
- name: Pulls the name from a parameter
- dependsOn: Needs a NIC resource to exist
- properties:
  - VMsize: How big a VM will we use?
    - Taken from a parameter
  - osProfile: Important OS settings
  - storageProfile
    - imageReference: What image we will use
    - osDisk: OS disk and how it's created
    - dataDisks: Additional data disks
  - networkProfile: Link to a NIC resource
  - diagnosticsProfile: Monitoring configuration

**Information about existing OS disks**
For the `storageProfile` we're using a marketplace image. Instead of this, we could use an existing VHD. First we'd create a Managed Disk from a VHD stored in a Storage Account, then choose **Attach** in the `storageProfile\osDisk` of the VM.

Two helpful links:
- MS Azure Github Quickstart Templates
- ARM Template Reference Documentation

Subscription and Services Layer

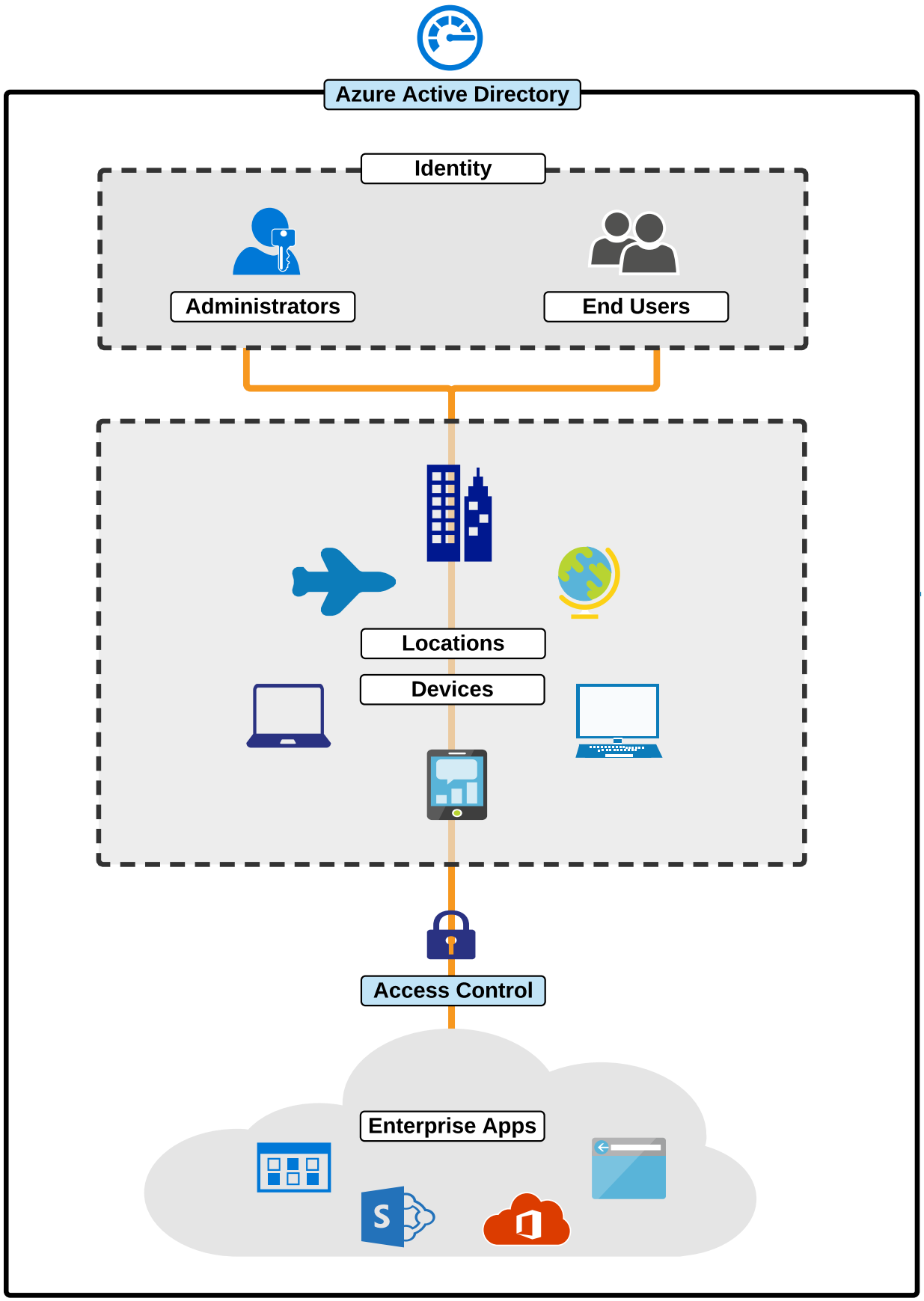
Physical and Networking Layer

## Identity and Access Management (IAM)

At the core of IAM within Azure (and other Microsoft services) is Azure Active Directory (AD).

This page depicts the real world scenario that Azure AD helps to address: managing many users accessing many applications from many locations.

Appendix



Subscription and Services Layer

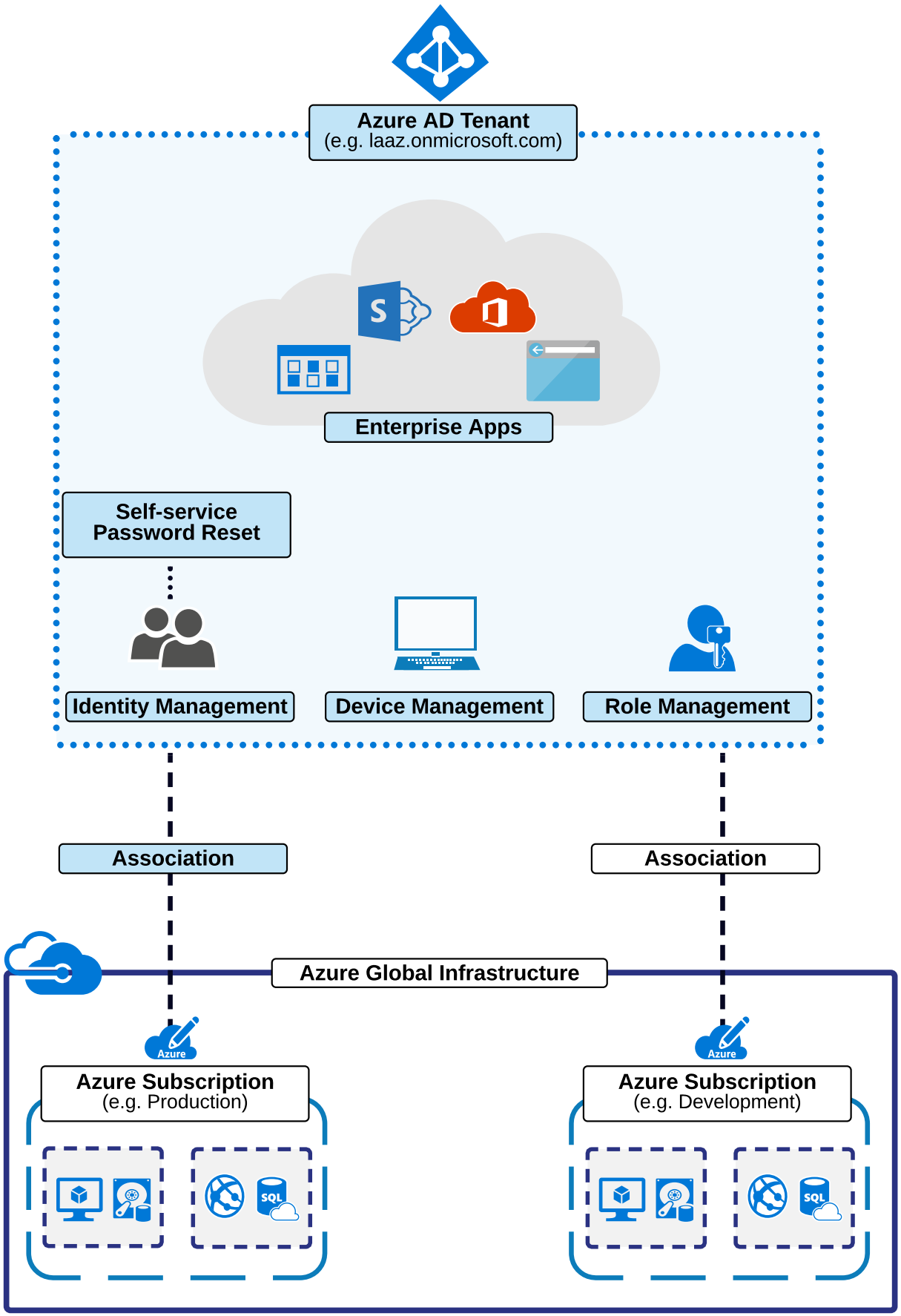
Physical and Networking Layer

## Azure Active Directory (AD)

Azure AD exists outside of Azure itself, and supports identity and access management for a range of Microsoft services.

This page helps demonstrate a range of Azure AD services, as well as the trust associated between Azure AD and Azure subscriptions.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

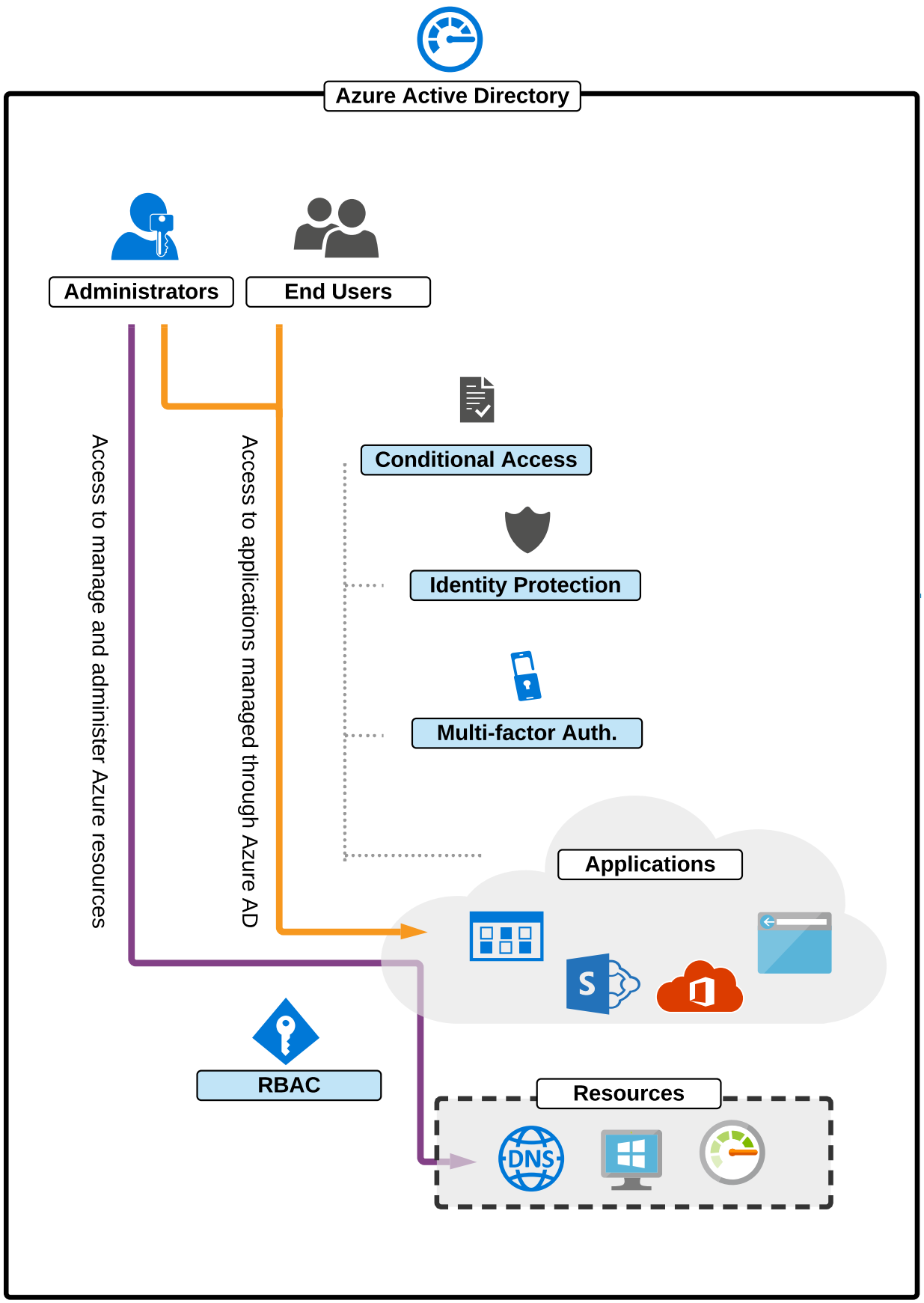
## Access Control

Azure AD provides features which help secure access to resources.

This page illustrates the main services available to protect Azure AD managed resources.

Role-based access control (RBAC) manages direct access to Azure resources.

Go Back Appendix



## Coming Soon

Close

The AZ-300 course is released **in preview**, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

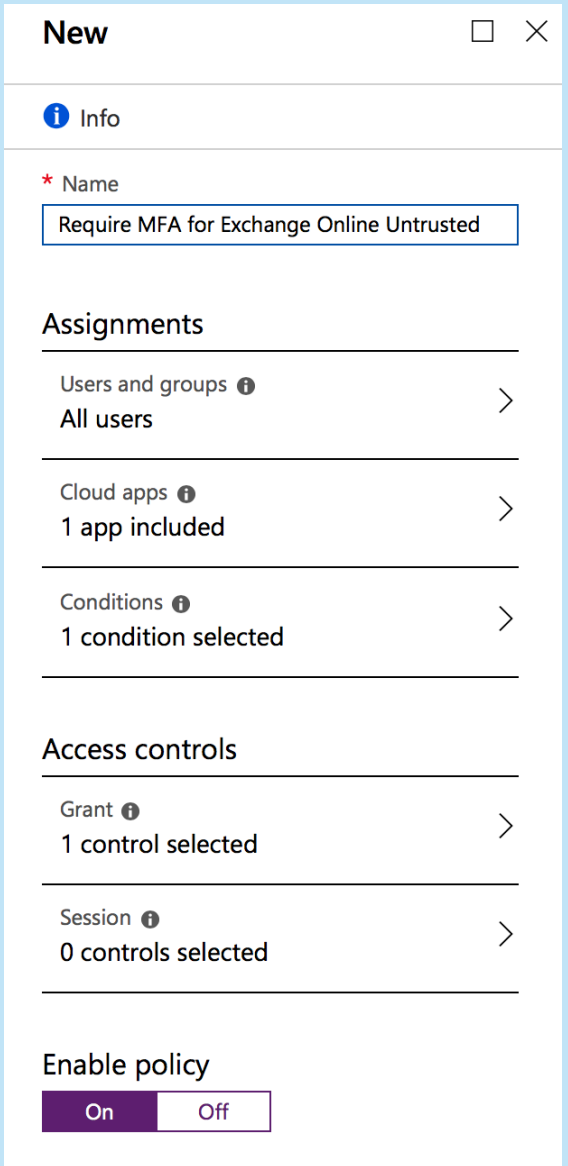
To continue, please click the close (X) button.

# Conditional Access

Close

Conditional Access provides the ability to configure a range of different access or deny rules to Azure AD managed resources, depending on a range of conditions.

For example, you could configure a Conditional Access Policy (CAP) that requires multi-factor authentication for all users when they sign into Microsoft Office 365 Exchange Online, unless they're signing in from a company's head-office.




**Creating a new Conditional Access Policy:**

The image to the left shows a new CAP being created within the Azure Portal. Below is an overview of key parameters of a CAP.

**Assignment:**
Each CAP defines a range of criteria which determine who, what, and under which circumstances a CAP will apply:

- Users and groups: Which users or groups the CAP applies to
- Cloud apps: Enterprise Applications, e.g. the Azure Portal
- Conditions: Risk (Azure AD ID protection), devices, locations

**Access controls:**
The access controls define whether access to the cloud app will be granted or blocked, and additional conditions which apply:

- Grant: Block, or grant (can also require other conditions like MFA)
- Session: Ability to enable limited experience access to an app

**Best practices and policy processing:**

- Block will always take precedence when multiple policies apply
- All access requirements must be met when multiple policies apply
- It is recommended that care is taken when using  all users/groups
- For more information refer to this Microsoft article

# Azure Multi-Factor Authentication (MFA)

Close

Microsoft provides a fully-managed multi-factor authentication (MFA) solution, which is a feature of Azure AD. This is often referred to as Azure MFA, or Cloud-based Azure MFA. This differs to MFA Server, which requires additional infrastructure to be deployed and managed.

Refer to this Microsoft article for a summary of differences between the two versions of MFA.

**The purpose of MFA:**
We use MFA as additional security to help authenticate a user. A common example of MFA is logging in to a bank with your credentials (username + password) as well as a code from an SMS. The terminology "multi-factor" refers to the need for multiple things being required to login: what you know (credentials) and what you have (access to your mobile).

**Important information about Azure MFA:**

- Azure MFA can be enabled in a number of ways, including:
  - Individually (through Azure cloud based MFA settings)
  - Conditionally (through Identity Protection, or Conditional Access Policies)
- Ideally a user should enroll and setup MFA *before* they are forced to authenticate with it, otherwise they may experience difficulties authenticating with apps (https://aka.ms/mfasetup)
- A range of verification options are available to users, including:
  - Call to a phone
  - Text message to a phone
  - Notification through mobile app
  - Verification code from mobile app or hardware token
- MFA can be bypassed:
  - According to specific criteria, through Conditional Access Policies
  - Using one-time bypass (for MFA Server) for a set amount of time
  - When the authentication request comes from a trusted IP

Close

# Role-based Access Control (RBAC)

While Conditional Access and Identity Protection are used to control access to Azure AD managed resources, RBAC is used to provide granular access to Azure resources themselves.

These roles can be assigned at the subscription, resource group, or resource level.

**Important information about RBAC:**

- Azure includes a range of over 70 **built-in roles** for controlling access to Azure resources:
  - Owner: Includes full access to the assigned resource(s) including rights to grant access to others
  - Contributor: Provides full access to the assigned resource(s) except rights to change permissions
  - Reader: Provides full view access to the assigned resource(s), but no ability to make changes

  *For more information, refer to the article on built-in roles for Azure resources.*

- If the built-in roles are not sufficient, **custom roles** can be created.
- For roles to take affect, they must be assigned:
  - Roles are assigned to an Azure AD user, group, or service principal
  - They must be assigned to something: a subscription, resource group, or resource
- Permissions granted through roles can be inherited (e.g. subscription owner owns all resources).
- Roles are versatile, and can be used for a range of access, for example:
  - Granting systems administrators access to all resources, except for billing
  - Granting service desk staff the ability to restart virtual machines only
  - Granting the finance team with access to billing, and support, but nothing else
- RBAC applies to Azure Resources only, but Conditional Access and Identity Protection can still block access to the Microsoft Azure Managent app, which includes all Azure management endpoints.

# Azure AD Roles

Close

There are three main types of administrative/security roles for Azure. These include classic subscription administrator roles, Azure role-based access control (RBAC) roles, and Azure AD administrator roles.

**Classic administrator roles:**

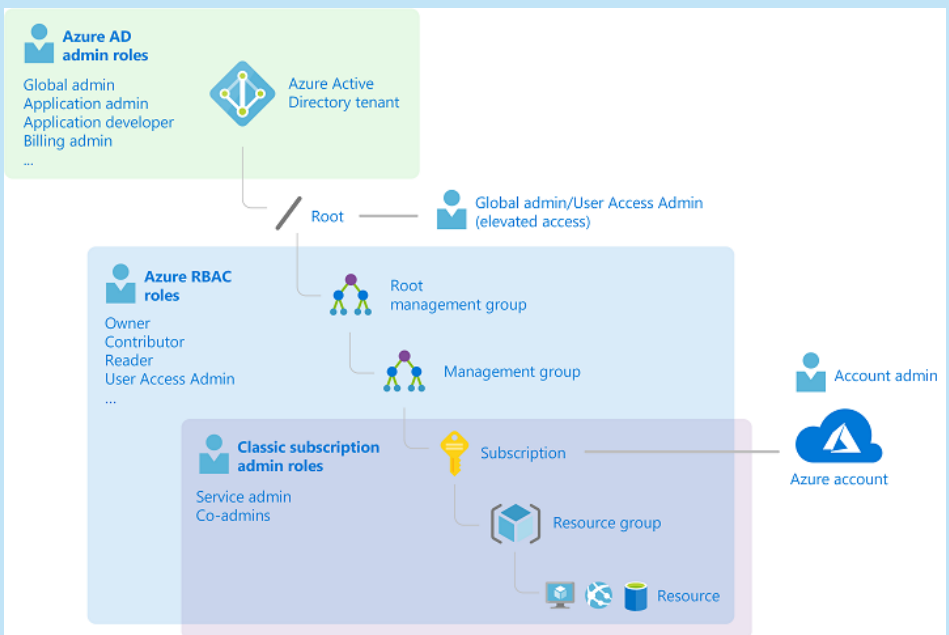
- Historical administrator: Roles which were originally used by Azure Service Manager
- Account administrator: Can manage/create/cancel subscriptions and change service administrator
- Service administrator: Can manage services within the Azure portal and co-administrators
- Co-administrator: Same permissions as service admin, but cannot manage classic admin roles

**RBAC roles**:

- Roles used to provide granular permissions to actual Azure resources
- For more information, take a look at RBAC roles within the Access Control page

**Azure AD administrator roles:**

- Roles for managing Azure AD itself (remember, the Azure AD tenant is separate to Azure)
- Includes a number of admin roles such as global, application, billing, etc
- Global admin provides full/unfettered access to Azure AD, and is typically required for enabling features




This diagram and more information can be found at this Microsoft article:
https://docs.microsoft.com/en-us/azure/role-based-access-control/rbac-and-directory-admin-roles

Close

# Enterprise Applications

Azure Active Directory (AD) provides the ability to manage applications, helping to support simplified usability (e.g. single sign-on) for users, as well as improved security for administrators.

**Why do applications matter in Azure AD?**
With a range of applications available to end-users in the cloud, providing a centralized access and identity management framework helps to improve security; administrators can centrally grant and revoke access to other applications for users, via Azure AD.

Security can be further enhanced through Azure AD Identity Protection and Conditional Access Policies.

**How do applications exist in Azure AD?**
Applications are typically defined within the application publishers own Azure AD directory, and can also then be referenced/accessed from a user's Azure AD directory. This is achieved through:

- Application objects - These can be considered the definition of the application within Azure AD.
- Service principals - Essentially an instance of an application, these are generally the link/reference to an application object.

For more information, refer to the following Microsoft article:
https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-how-applications-are-added

Close

# Self Service Password Reset (SSPR)

Self Service Password Reset (SSPR) is an Azure AD feature which gives end-users the ability to reset their password without having to contact an Azure AD administrator. End-users can enroll in SSPR and use a range of authentication methods to verify their identity and reset their password.

**Important information about SSPR:**

- SSPR is enabled within the Azure AD settings for your tenant:
  - Navigate to Azure AD > Password Reset, Properties,
  - Configure SSPR to be enabled for None, Selected (groups), or All.
- Communication is an important step of enabling SSPR, as users must complete a registration process.
- There are a range of important links which can be accessed by users:
  - Azure AD password reset registration portal - https://aka.ms/ssprsetup,
  - Azure AD password reset portal - https://aka.ms/sspr,
  - Azure AD password change portal - https://account.activedirectory.windowsazure.com/ChangePassword.aspx.
- Available authentication methods include:
  - Password
  - Security questions (including predefined and custom questions)
  - Email address
  - Microsoft Authenticator app (preview)
  - OATH Hardware token
  - SMS or voice call

# Azure AD Identity Protection - Overview



Azure Active Directory (AD) Identity Protection is a service provided by Microsoft. It is used to intelligently identify risks and vulnerabilities affecting your Azure AD identities. Identity Protection is powered by adaptive machine learning which detects anomalies and suspicious incidents relating to identity.

**To enable Azure AD Identity Protection:**

- You must have Azure AD Premium P2 licensing
- You must also create Azure AD Identity Protection, as a new resource

**A summary of Identity Protection capabilities is as follows:**

- Detect vulnerabilities and risky accounts based on various patterns and heuristics
- Policies which can be configured to:
  - Mitigate risky sign-ins by blocking sign-ins entirely or requiring multi-factor authentication
  - Block or secure risky user accounts
  - Require users to register for multi-factor authentication

**Policies within Identity Protection:**

It is possible to configure policies within Identity Protection itself, and/or Conditional Access Policies. The policies within Identity Protection which can be configured are as follows:

- Multi-factor authentication (MFA) registration policy:
  - Policy for enforcing user MFA registration,
  - Allows more granular control over registration compared to standard Azure MFA settings
- User risk policy:
  - Policies applying to accounts which Identity Protection has identified as being at risk
  - Allows or denies access based on risk level, and can enforce a password change
- Sign-in risk policy:
  - Policies get applied in real-time, during sign-in, and can therefore be used to prevent sign-in.
  - They allow or deny access based on risk level, and can enforce the use of MFA.
  - These apply to browser and modern-auth traffic, but not apps which use older security protocols.

# Azure AD Identity Protection - Risk Events

Close

Overview | Risk Events | Vulnerabilities

At the heart of Azure AD Identity Protection is the evaluation and identification of risk events. For example, Azure AD might consider a sign-in risky if it's coming from a country where the user ordinarily does not sign-in from.

**Below are a list of the risk events currently detected:**

- Users with leaked credentials
- Sign-ins from anonymous IP addresses
- Impossible travel to atypical locations (like a sign-in from India, then another in 5 minutes from America)
- Sign-ins from infected devices
- Sign-ins from IP addresses with suspicious activity
- Sign-ins from unfamiliar locations

Each of these risk events has an associated **risk level (high, medium, or low)** which can be used for reporting purposes or within Conditional Access Policies to mitigate risk.

| RISK LEVEL | DETECTION TYPE | RISK EVENT TYPE | RISK EVENTS CLOSED | LAST UPDATED (UTC) |
|---|---|---|---|---|
| High | Offline | Users with leaked credentials | 44 of 45 | 12/7/2016 1:04 AM |
| Medium | Real-time | Sign-ins from anonymous IP addresses | 76 of 78 | 1/17/2017 2:44 PM |
| Medium | Offline | Impossible travels to atypical locations | 11 of 14 | 1/17/2017 2:44 PM |
| Medium | Real-time | Sign-in from unfamiliar location | 0 of 1 | 11/15/2016 7:18 PM |
| Low | Offline | Sign-ins from infected devices | 76 of 78 | 1/17/2017 2:44 PM |

The above diagram is from the following Microsoft article, which includes additional details on risk events: https://docs.microsoft.com/en-us/azure/active-directory/reports-monitoring/concept-risk-events

# Azure AD Identity Protection - Vulnerabilities

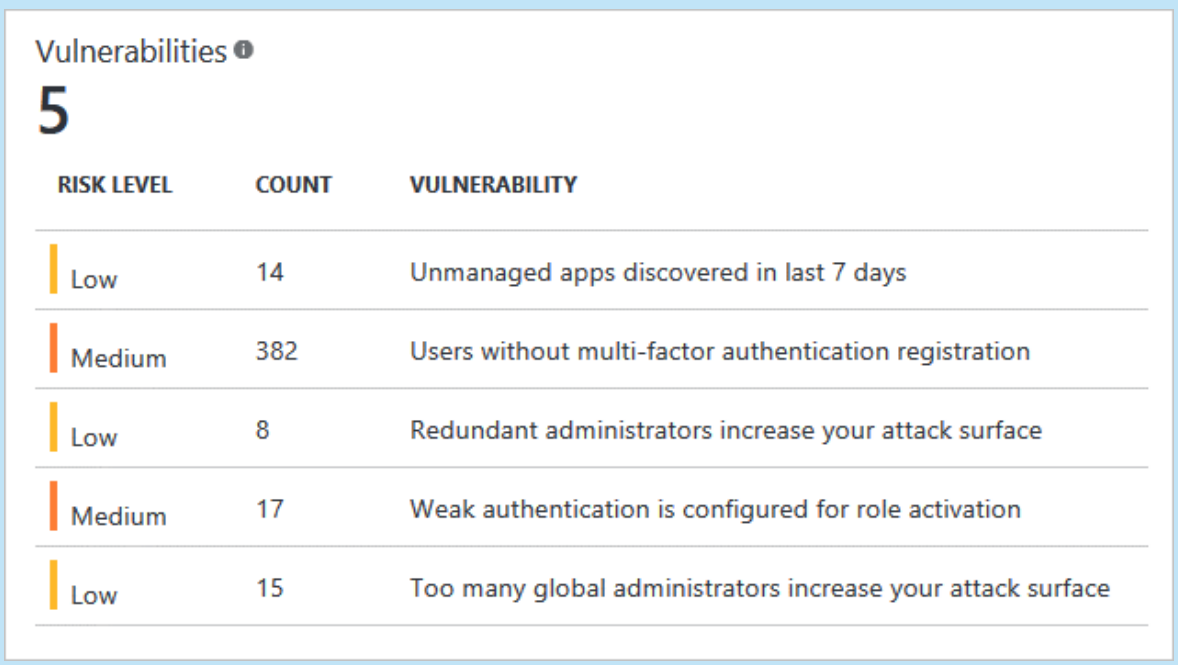
Close

Overview | Risk Events | Vulnerabilities

Azure AD Identity Protection evaluates your environment for any vulnerabilities which may be exploited by an attacker. These vulnerabilities are then reported for your investigation and remediation.

**Below is a summary of the vulnerabilities which can be reported by Identity Protection:**

- Multi-factor authentication registration not configured
- Unmanaged cloud apps (requires Cloud Discovery to be deployed)
- Security Alerts identified through the Azure AD Privileged Identity Management service (if enabled)



Vulnerabilities

5

| RISK LEVEL | COUNT | VULNERABILITY |
|---|---|---|
| Low | 14 | Unmanaged apps discovered in last 7 days |
| Medium | 382 | Users without multi-factor authentication registration |
| Low | 8 | Redundant administrators increase your attack surface |
| Medium | 17 | Weak authentication is configured for role activation |
| Low | 15 | Too many global administrators increase your attack surface |

The above diagram is from the following Microsoft article, which includes additional details on vulnerabilities: https://docs.microsoft.com/en-us/azure/active-directory/identity-protection/vulnerabilities

# Azure Active Directory (AD)



Azure Active Directory (AD) is a cloud-first identity and access management (IAM) service. Azure AD is very similar in purpose to Microsoft's traditional IAM solution, Active Directory.

Azure AD supports a range of objects which are often used with other Azure AD services to support functionality, such as access control, single sign-on, etc.

**An example of the types of resources and objects you will find in Azure AD are as follows:**

- User accounts and credentials
- Profile information (location, phone number, address, etc.)
- Security groups
- Enterprise applications
- Registered devices

**Some important facts about Azure AD are:**

- Several Microsoft services leverage Azure AD, including Azure and Microsoft Office 365.
- The Azure AD service exists outside of Azure.
- For an Azure subscription to leverage an Azure tenant, the two must be **associated**.
- A range of licenses are available for Azure AD, and these determine the functionality available.
- Azure AD is different than Microsoft AD, though the two can be integrated using Azure AD Connect.

# Azure AD - Domain Names



Each Azure AD tenant is created with an initial domain name of *domainname*.onmicrosoft.com. This domain name is critical to Azure AD, and is associated with objects such as users (e.g. jsmith@domainname.onmicrosoft.com).

**Important information about Azure AD domain names:**

- The initial domain name cannot be deleted or changed
- Azure AD tenants support custom domain names, which:
  - Can be added through the Azure portal (Azure AD > Custom domain names),
  - Must be verified through DNS records (either TXT or MX record types).

# Azure AD - Association and Subscription Trust



All Azure subscriptions are associated with a single Azure AD tenant. This provides the underlying identity and access management foundation.

It is important to understand that:

- An Azure subscription can only be associated with a single Azure AD tenant.
- An Azure AD tenant can be associated with multiple subscriptions.
- This association is what allows identity and access to be managed through:
  - Role-based access control - Azure AD identities granted permissions to Azure resources
  - Single sign-on - Azure AD identities granted permissions to access enterprise applications
  - Access control - Azure AD used to store users, devices, applications, and provide features with control access based on one or more of these properties

# Device Management - Overview

Close

Overview | ESR

Azure AD provides the foundation for the ability to manage devices from the cloud. This capability ranges from simple device registration on through to full mobile device and application management (MDM/MAM).

Device registration, restrictions, and other settings can be found within the Azure AD settings:

- Navigate to Azure AD > Devices > Device Settings, and select whether devices may be registered

There are three main methods for registering devices with Azure AD:

- Azure AD registered devices:
  - Typically used for personal devices not owned by the organization (bring your own device; BYOD)
  - Registers the object in Azure AD, providing an identity for the device
  - Supports access to an organization's Azure AD controlled resources
- Azure AD joined devices
  - Typically used for devices that are owned by an organization which does not use on-prem AD
  - Changes the local state of the device so that Azure AD logins can be used to log into the device
  - Provides access to single sign-on to Azure AD managed resources, **enterprise state roam (ESR)**, restriction of access based on device compliance, and other benefits
- Hybrid Azure AD joined devices
  - Typically used when an organization already uses, and has devices joined to, on-prem AD
  - Supports features of Azure AD join such as single sign-on and ESR

More information can be found from this Microsoft article:
https://docs.microsoft.com/en-us/azure/active-directory/devices/overview

# Device Management - Enterprise State Roam (ESR)

Close

Overview | ESR

Enterprise State Roam (ESR) is an Azure AD feature which provides the ability to synchronize user and application settings data to the cloud. This provides users with a unified experience across their Windows devices; settings configured on one Azure AD managed device will be the same on others.

**Important information about Azure AD Enterprise State Roam:**

- ESR can be enabled through the Azure Portal (Azure AD > Devices > Enterprise State Roaming)
- Data for ESR is:
  - Hosted in one or more Azure regions aligned to the country/region of the Azure AD tenant,
  - Retained 90 to 180 days after explicit deletion (like if the user or directory is deleted)

Subscription and Services Layer

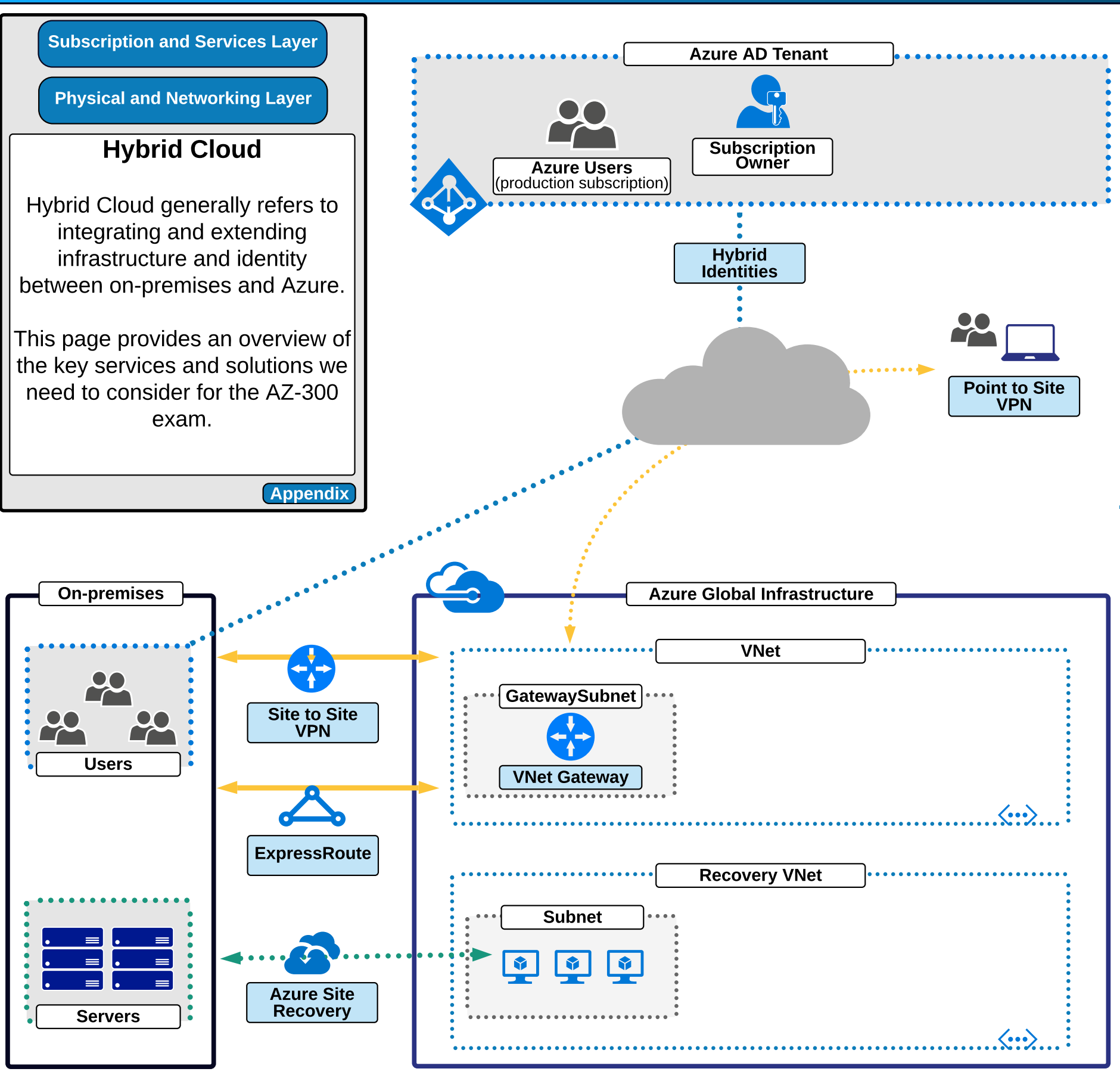
Physical and Networking Layer

## Hybrid Cloud

Hybrid Cloud generally refers to integrating and extending infrastructure and identity between on-premises and Azure.

This page provides an overview of the key services and solutions we need to consider for the AZ-300 exam.

Appendix



Subscription and Services Layer

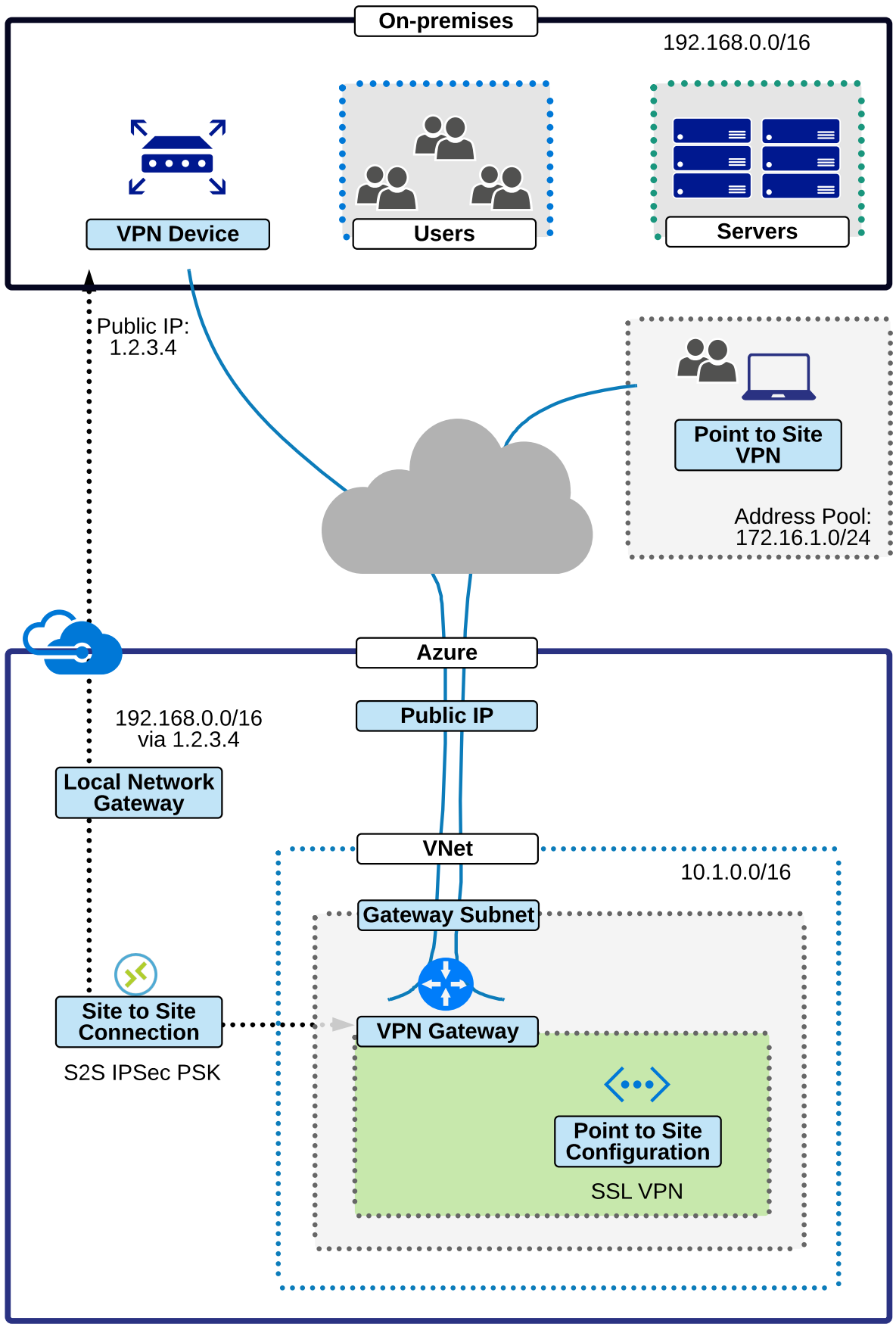
Physical and Networking Layer

## VPN Gateway

We can connect an on-premises network to an Azure Virtual Network using a VPN Gateway.

This type of connectivity utilizes an encrypted tunnel over the public Internet.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

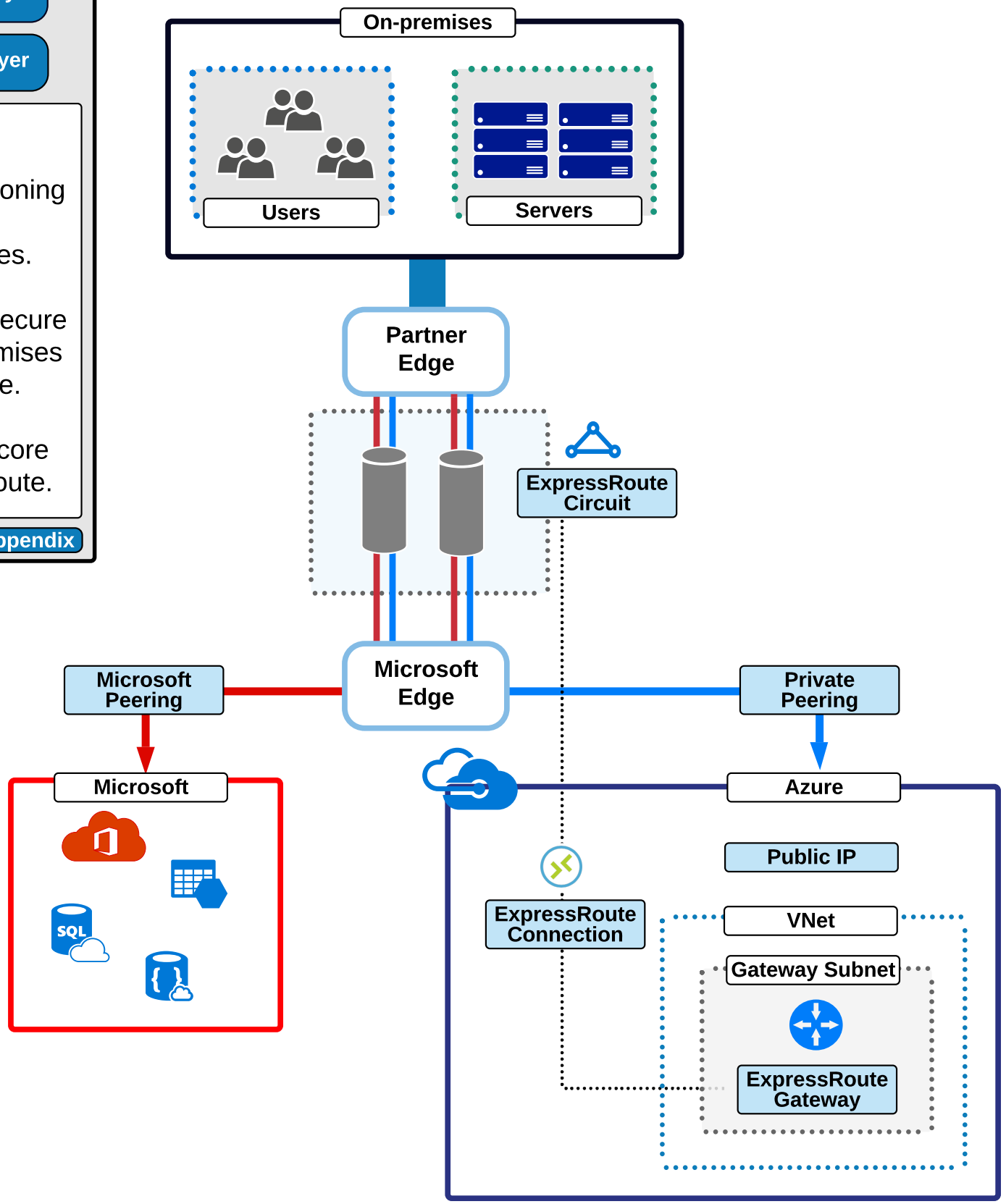
## ExpressRoute

Many organizations transitioning to cloud already have infrastructure on-premises.

ExpressRoute provides a secure way for connecting on-premises infrastructure with Azure.

This diagram depicts the core components of ExpressRoute.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

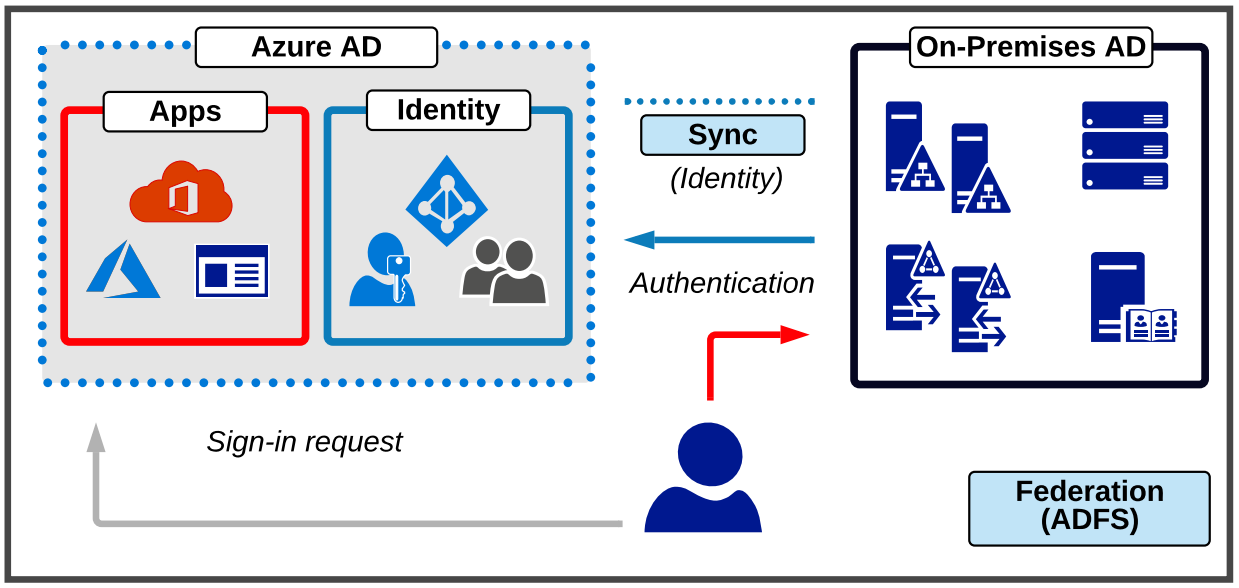
## Hybrid Identity

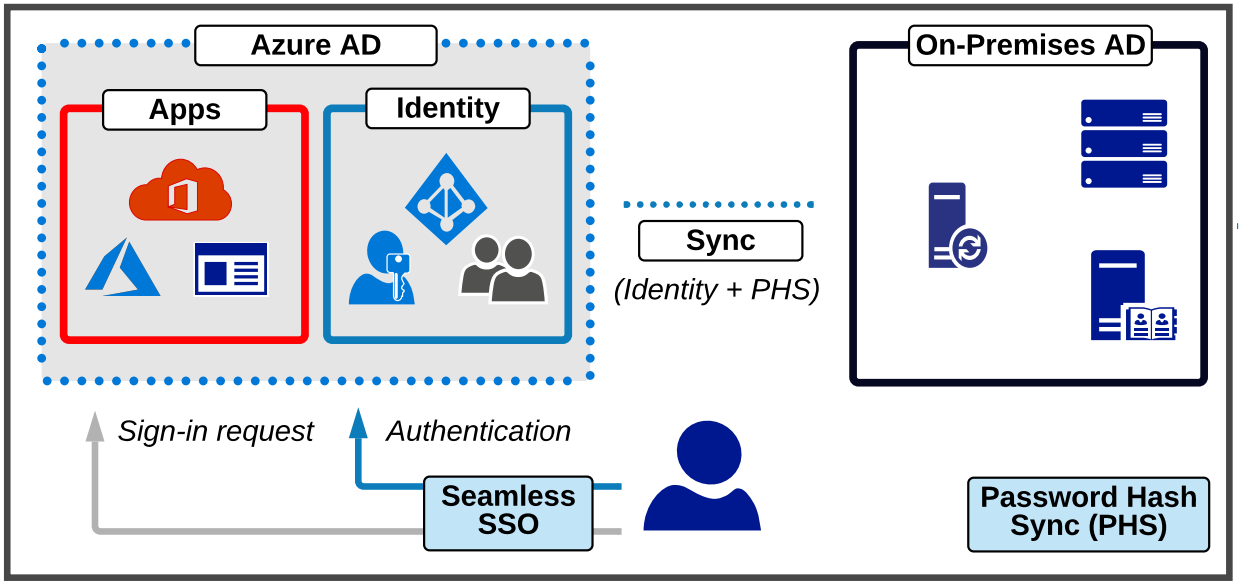
Many organizations moving to the cloud already manage identity on-premises.

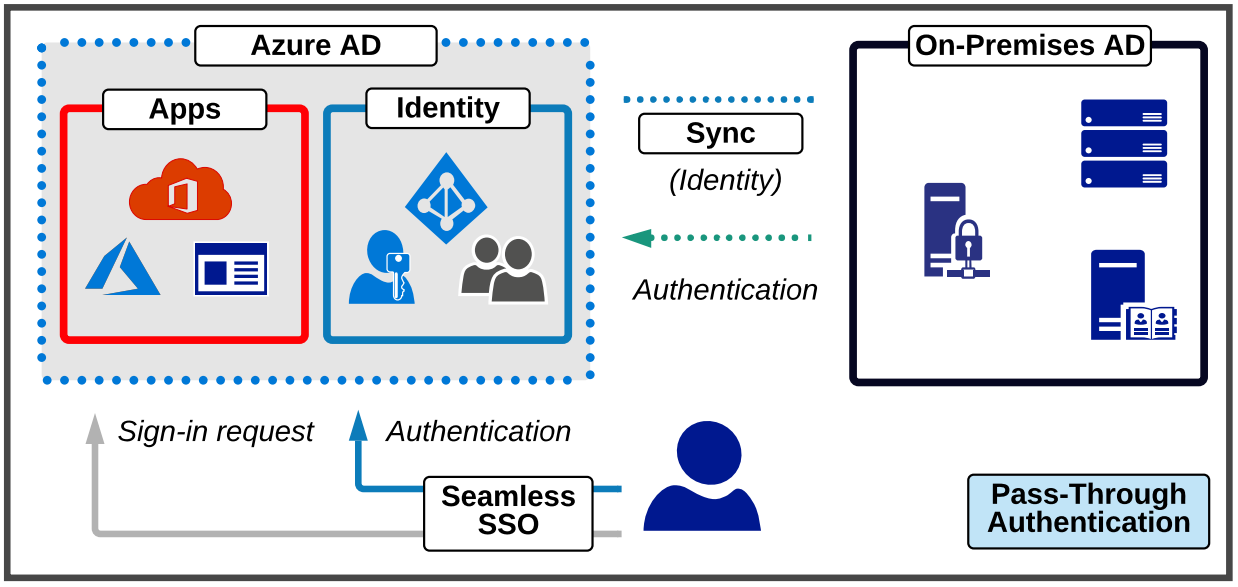
With Azure AD Connect you can extend this identity in to the cloud (hybrid identity) and allow users to access cloud applications with existing credentials (single sign-on).

Go Back

Appendix



Subscription and Services Layer

Physical and Networking Layer

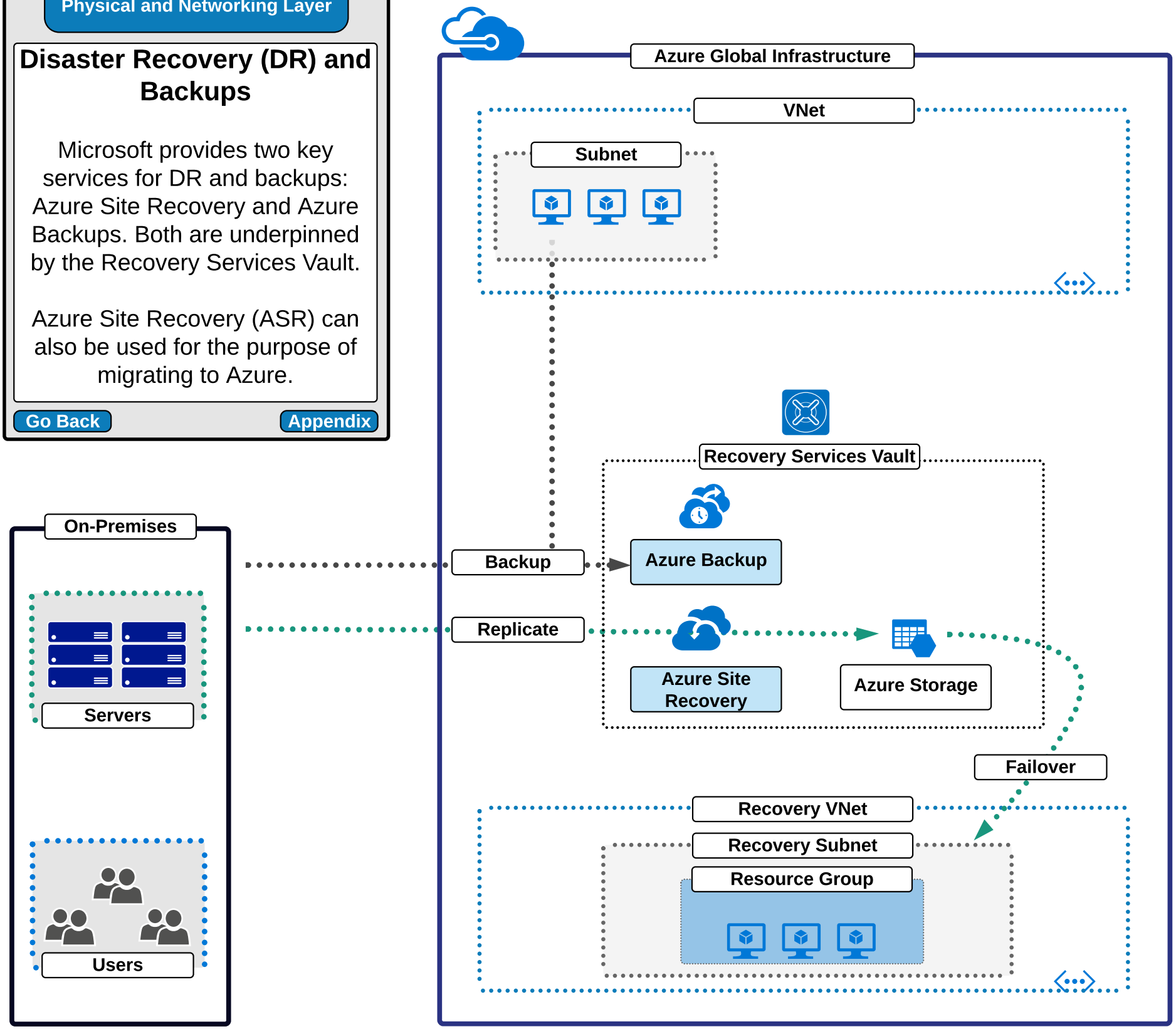
## Disaster Recovery (DR) and Backups

Microsoft provides two key services for DR and backups: Azure Site Recovery and Azure Backups. Both are underpinned by the Recovery Services Vault.

Azure Site Recovery (ASR) can also be used for the purpose of migrating to Azure.

Go Back

Appendix



## Coming Soon

Close

The AZ-300 course is released in ***preview***, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

To continue, please click the close (X) button.

# Azure AD Connect

Close

Azure Active Directory (AD) Connect is the underlying Microsoft tool used to deploy, configure, manage, and monitor hybrid identity between on-premises AD and Azure AD.

Azure AD Connect is supported on Server 2012 R2 and up.

**Key features of the tool:**

- Provides the ability to configure and deploy the following hybrid identity solutions:
  - Password hash synchronization (PHS)
  - Pass-through authentication (PTA)
  - Federation integration including AD Federation Services
- Synchronization of users, groups, and other objects between on-premises AD and Azure AD
- Health monitoring, providing monitoring data which is visible within the Azure Portal

**Important information about the synchronization service:**

- **Staging** mode allows for Disaster recovery of Azure AD Connect:
  - Azure AD Connect is installed to a separate server ideally in a different location to the first.
  - During the configuration of Azure AD Connect, staging mode is enabled.
  - Staging mode must be disabled for synchronization and hybrid identity functionality to be enabled.
- The following is a summary of some key management operations:
  - To check the status of the synchronization service with PowerShell:
    .
  - The Synchronization Service Manager GUI tool supports configuration and monitoring of synchronization operations.
  - Sync operations can be triggered with PowerShell: .
- Note that by default, sync operations will operate every 30 minutes.

Close

# Seamless Single Sign-On (Seamless SSO)

When Azure Active Directory (AD) Connect is deployed with either pass-through authentication (PTA) or password hash synchronization (PHS), it is possible to choose a 'single sign-on' option during configuration.

This option enables what is often referred to as seamless SSO, a feature intended to improve the user sign-in experience. In other words, it reduces the number of prompts for credentials.

**Important information about seamless SSO:**

- This mode of SSO is not supported with AD Federation Services.
- Users are automatically signed in to both on-premises and cloud-based applications.
- Users do not have to enter their passwords repeatedly.
- Underlying configuration of both Azure AD and on-premises AD is managed by Azure AD Connect.

More information can be found from the following Microsoft article:
https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-sso

Anyone interested in the underlying implementation of seamless SSO can refer to the following Microsoft article: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-sso-how-it-works

Close

# Azure Backup

Azure Backup provides the ability to back up (protect) and restore data from a range of sources, to a Recovery Services Vault (RSV) within the Microsoft cloud.

**To summarize of the features provided by Azure Backup:**

- Backup storage and scaling is managed by Microsoft.
- Azure Backup provides support for various clients and backup goals:
  - Support for cloud virtual machines including AWS (and native support for Azure VMs)
  - Support for Linux and Windows (note that not all backup tools support Linux)
  - Protection of files, system state, and applications, VMs (not all tools support all protection types)
- There are a number of tools/components which can be used in different scenarios:
  - Azure Backup (MARS) agent - For physical or virtual Windows OS, direct to a RSV
  - System Center DPM - For Linux and Windows OS, to RSV, tape, or locally attached disks
  - Azure Backup Server - Similar to DPM but relying on an Azure subscription
  - Azure IaaS VM Backup - With no specific agent install required, direct to RSV

Refer to the Introduction to Azure Backup article for more information on backup scenarios and tools.

# Azure Site Recovery (ASR) - Overview

Close

Overview | Migration

Azure Site Recovery (ASR) can be considered "Disaster Recovery as a Service." The functionality provided by ASR is centered on providing an two core services:

- Replication of servers from one location (source) to a separate location (target)
- Capability to failover to the target destination in the event of a disaster occurring at the source

**The main features of ASR are:**

- Disaster recovery: Failover from one site to another, for example:
  - From on-premises to Azure
  - From on-premises to another site you manage
- Support for different sources, including physical, VMware, and Hyper-V
- While ASR is focused on disaster recovery, the capabilities can also be used for migration:
  - Migration uses the same technology, but failover is performed one way only without failing back
  - Migration to Azure from physical servers, vCenter, Hyper-V, AWS, or even Azure using ASR

**Key components and tools of ASR and their purpose:**

- Main server components for physical, AWS, or VMware source environments:
  - Configuration server: Coordinates communication between on-premises and Azure
  - Process server: Installed by default on the config server, and sends data to Azure Storage intelligently
  - Mobility Service: Coordinates replication between on-premises and Azure
  - Master target server: Used for failback (and so not used in migration scenarios)
- Main server components for Hyper-V source environments:
  - Azure Site Recovery Provider: Orchestrates replication and failover
  - Recovery Services Agent: Performs replication with Azure from each Hyper-V host or cluster node
- The Azure Site Recovery Deployment Planner provides the following:
  - Support for both VMware and Hyper-V
  - Command line tool ()
  - Helps provide details for:
    - Compatibility assessment
    - Network bandwidth needs
    - Azure infrastructure requirements
    - On-premises infrastructure requirements
    - Estimated disaster recovery cost to Azure

# Azure Site Recovery - Overview of Migration Setup

Close

Overview | Migration

The following is an overview of the steps and some important information relating to configuring Azure Site Recovery for the purpose of migrating from on-premises to Azure.

**Azure (target environment) preparation:**

- Create a Recovery Services Vault
- Configure target Storage Account and Virtual Network (VNet)

**On-premises (source environment) preparation:**

- For physical, AWS, and VMware source environments:
  - Deploy configuration server (must be Server 2012 R2 or Server 2016)
    - Simplified with VMWare as Microsoft provides an Open Virtual Application (OVA) template
    - Configuration server must be registered using the ASR Vault Credentials during install
  - Process Server is included with the Configuration Server
  - Mobility Service requires credentials in order to be deployed to source servers
- For Hyper-V source environments:
  - Azure Site Recovery Provider and Recovery Services Agent must be deployed

**Configure replication:**

- Setup a replication policy for:
  - Recovery Point Objectives (RPO): How often recovery points are created
  - Recovery point retention between 0-72 hours: How many recovery points to retain (0 = latest only)
  - App consistent snapshot frequency: How often to perform an app consistent snapshot (VMs)
- Associate the replication policy with the Configuration Server
- Enable replication: source, target, servers, settings (credentials, disk exclusions), policy

**Perform migration:**

- Perform a failover
  - Navigate to the Recovery Services Vault > Replicated Items > Select the VM > Failover

# Federation - Overview

Close

Overview | Deployment

When an on-premises directory is federated with Azure Active Directory, a trust gets established which provides authentication (confirming you are who you say you are) and authorization (determining whether or not you are allowed access).

Using Azure AD Connect, we can configure federation with either Active Directory Federation Services (ADFS) or PingFederate. With federated identity, all user authentication occurs on-premises.

**Important notes about Azure AD Connect and AD FS:**

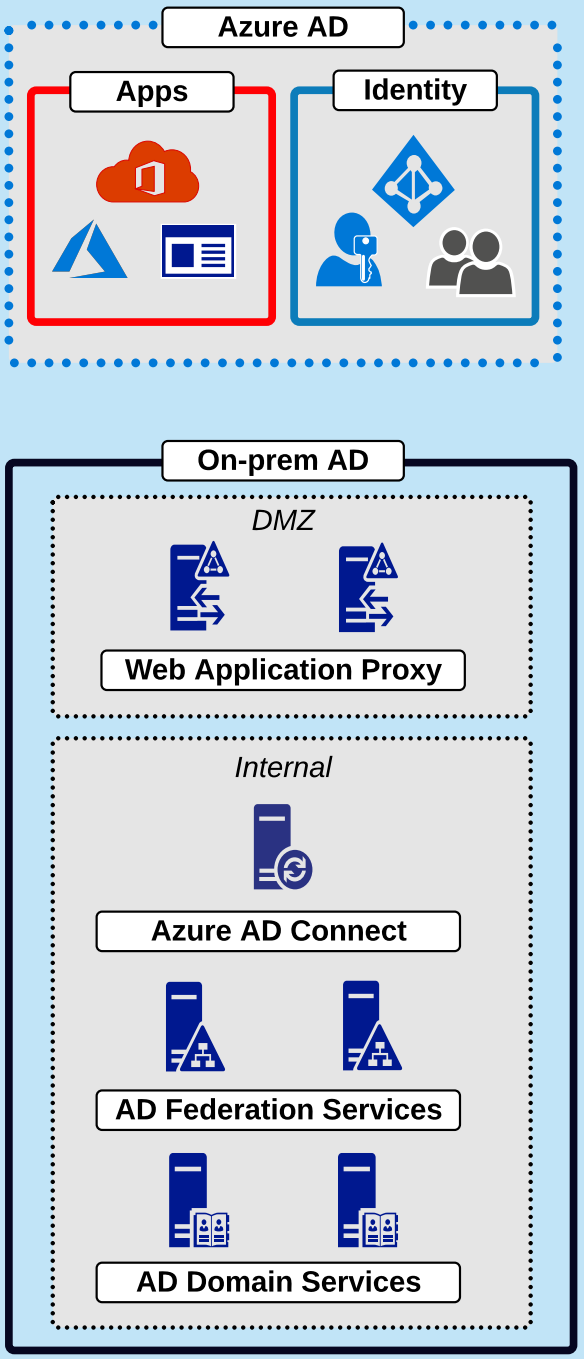
- The main benefits:
  - Synchronization of users, contacts, and group accounts between on-premises and Azure AD
  - Supports Office 365 hybrid (really common use-case)
  - Enables users to sign in and access cloud services/apps using on-premises credentials
  - Does not require password hashes to be stored in the cloud
  - Supports an array of third-party and on-premises multi-factor authentication solutions
  - Supports smart card authentication
  - Allows the display of password expiry notifications in the Office Portal and Windows 10 desktop
  - Supports all on-premises account policies (e.g. expiry, hours logged in, etc. as on-premises sign-in occurs)
- Important considerations, when compared to Password Hash Sync or Pass-through Authentication:
  - Requires more infrastructure,
  - Is more complex to configure and maintain
  - Does not support seamless single sign-on
- For a detailed comparison of the Azure AD Connect options, refer to this Microsoft article

# ADFS Deployment Overview

Close

Overview | Deployment

Below is an overview of the configuration and infrastructure requirements for ADFS federated identity.




**Configuration overview:**

- Custom domain added and verified in Azure AD
- Azure AD Connect installed on Server 2012 R2 or later:
  - Using ADFS hybrid identity option:
    - One or more ADFS servers
    - One or more ADFS Web Application Proxies
    - SSL certificate for the federation service name
  - Azure AD Connect deployed including:
    - Azure AD Connect Synchronization and Health.

The diagram to the left depicts a standard deployment.

# Password Hash Sync (PHS) - Overview



Password hash synchronization (PHS) is underpinned by the Azure AD Connect synchronization service. PHS synchronizes a hash, of the hash, of a user's on-premises password to Azure Active Directory (AD).

Using Azure AD Connect, we can configure PHS so that all user authentication occurs in Azure AD. PHS can optionally be configured as a backup for ADFS.

Azure AD Connect *express* install is a standard deployment of Password Hash Sync.

**Important notes about Azure AD Connect and PHS:**

- The main benefits:
  - Synchronizes of users, contacts, and group accounts between on-premises and Azure AD
  - Supports Office 365 hybrid (really common use-case)
  - Enables users to sign in and access cloud services/apps using on-premises credentials
- Important considerations:
  - When compared to pass-through authentication and ADFS, PHS provides the least features
  - Multi-factor authentication (MFA) with PHS is only possible using Azure AD MFA
  - Some organizations have security restrictions which prevent passwords being stored in the cloud
  - PHS changes the behavior of password policies and expiry in Azure AD:
    - Password complexity rules from on-premises take precedence
    - Password expiry is set to never expire

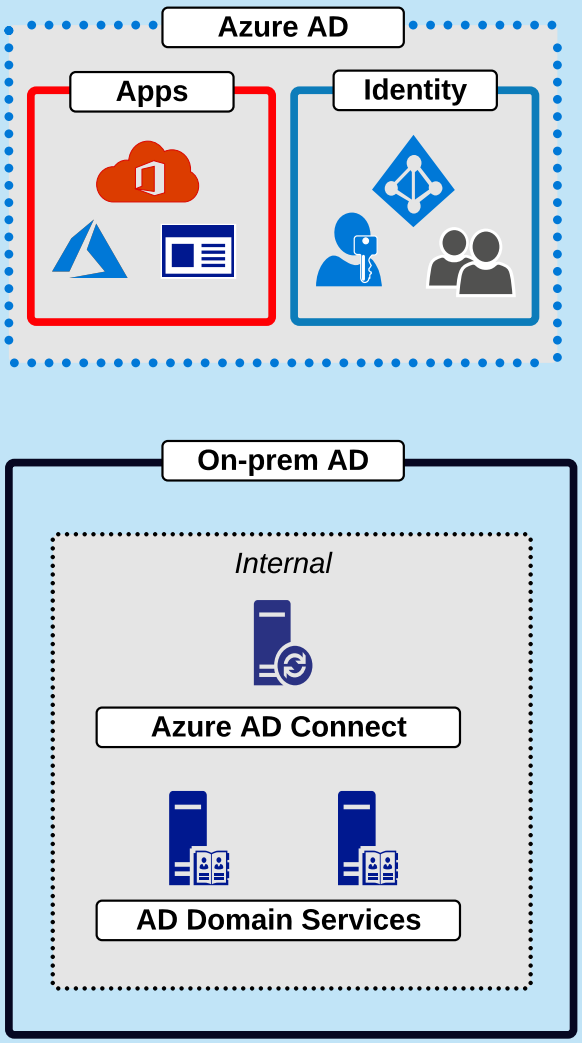
For a detailed comparison of the Azure AD Connect options, refer to this Microsoft article.

Close

# PHS Deployment Overview

Overview | Deployment

Below is an overview of the configuration and infrastructure requirements for PHS.




**Configuration overview:**

- Custom domain added and verified in Azure AD
- Azure AD Connect installed to Server 2012 R2 or later:
  - Uses the password hash synchronization option, which is part of the Azure AD Connect deployment
  - Single sign-on enabled (optional but recommended)
  - Includes Azure AD Connect Synchronization and Health

The diagram to the left depicts a standard deployment.

# Pass-Through Authentication (PTA) - Overview

Close

Overview | Deployment

Pass-through authentication (PTA) is an alternative to password hash synchronization (PHS). PTA provides the same seamless single sign-on experience as PHS, but offers additional security benefits.

With Azure AD Connect configured for PTA, we have a unique authentication experience that offers a number of the benefits of both PHS and AD Federation Services. All PTA user authentication occurs on-premises, but through an outbound-only connection from an on-premises authentication agent.

**Important notes about Azure AD Connect and PTA:**

- The main benefits:
  - Synchronization of users, contacts, and group accounts between on-premises and Azure AD
  - Supports Office 365 hybrid (really common use-case)
  - Enables users to sign in and access cloud services/apps using on-premises credentials
  - Does not require password hashes to be stored in the cloud
  - Only requires outbound connectivity from the on-premises Authentication Agents
  - All on-premises account policies enforced at the time of sign in (e.g. expiry, logon hours, etc.)
- Important considerations:
  - On-premises multi-factor authentication (MFA) solutions are not supported with PTA.
  - PTA is not integrated with Azure AD Connect Health.

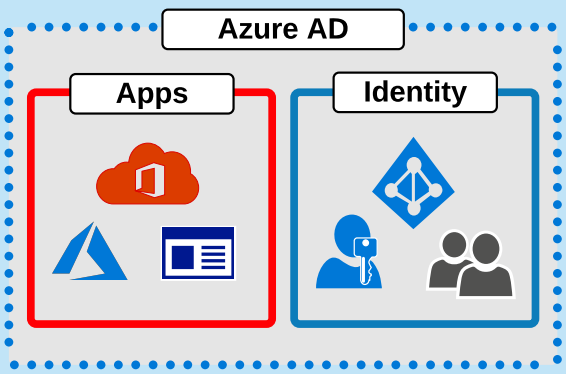
For a detailed comparison of the Azure AD Connect options, refer to this Microsoft article.

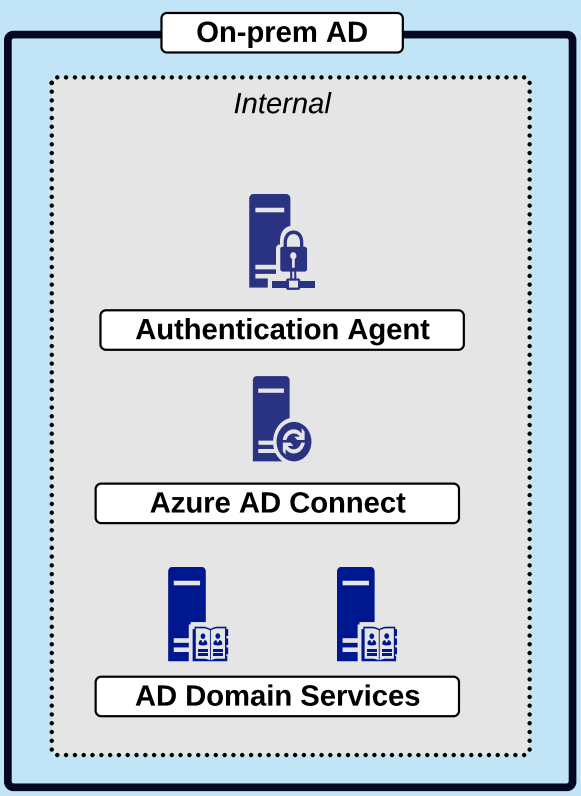
# PTA Deployment Overview

Close

Overview | Deployment

Below is an overview of the configuration and infrastructure requirements for Pass-through authentication.




**Configuration overview:**

- Custom domain added and verified in Azure AD
- Azure AD Connect installed to Server 2012 R2 or later:
  - Using the pass-through authentication option, which is part of the Azure AD Connect deployment
  - Single sign-on enabled (optional but recommended)
  - Includes Azure AD Connect Synchronization and Health
- Authentication Agent installed to one or more servers

The diagram to the left depicts a standard deployment.

**Information about the architecture of PTA:**

- PTA is the newest authentication option for Azure AD Connect
- Improved security is achieved using an Azure Service Bus queue

# ExpressRoute - Overview

Close



Like a site-to-site VPN, ExpressRoute can provide connectivity between an Azure Virtual Network (VNet) and networks you own outside of Azure, your on-premises or datacenter environment for example.

**Main benefits of ExpressRoute:**

- Unlike a site-to-site VPN, ExpressRoute does not go out over the public Internet.
- Connectivity is provided through an ExpressRoute connectivity provider which yields:
  - Greater security
  - More reliable connectivity (including redundant ExpressRoute circuits as standard)
  - Direct connectivity with Microsoft services

**Important notes about ExpressRoute:**

- There are two main types of ExpressRoute connections (routing domains):
  - Azure Private Peering - Connectivity between an on-premises network and a VNet
  - Microsoft Peering - Connectivity to Microsoft Online Services (O365, Azure PaaS, Dynamics 365, and a range of others).
- ExpressRoute and site-to-site VPNs can coexist, but requires non-basic SKU and route-based VPN.
- An ExpressRoute Cirtcuit can connect to multiple VNets, both inside and outside of the subscription.

**At a high level, the key steps in the configuration of ExpressRoute are:**

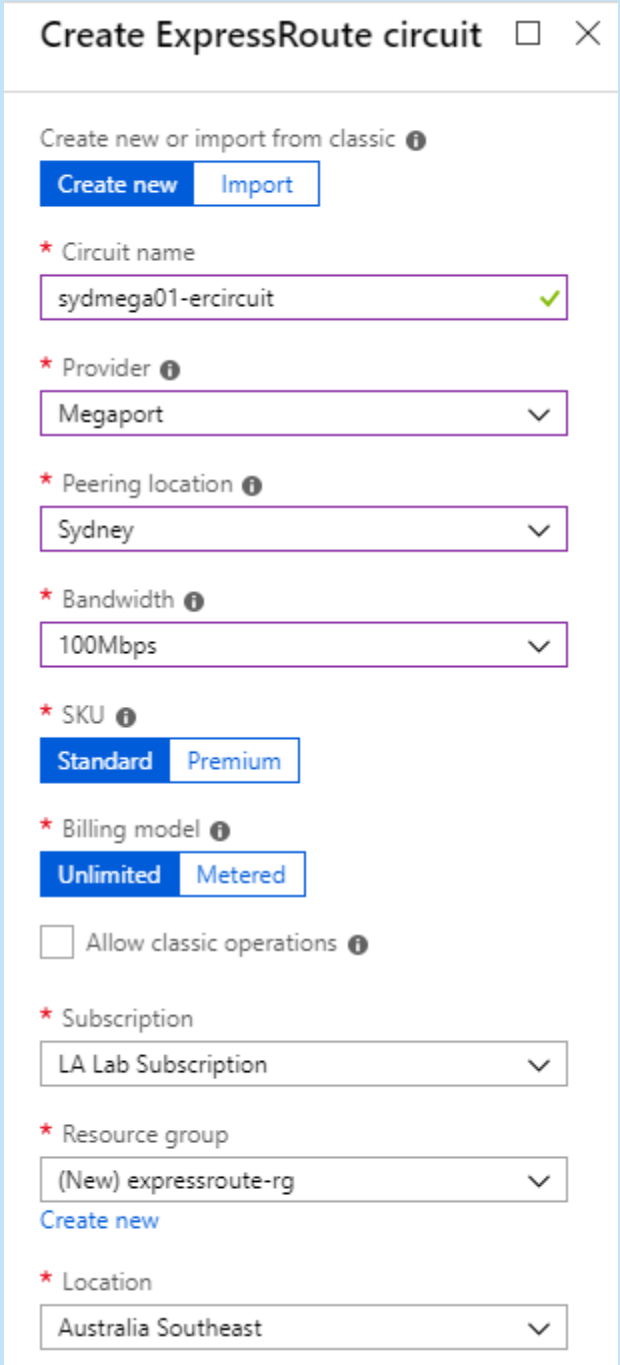
- Create the ExpressRoute Circuit.
- Work with your connectivity provider to ensure the ExpressRoute Circuit is provisioned.
- Configure the ExpressRoute routing domains (peerings).
- Configure the Virtual Network (VNet) Gateway.

# ExpressRoute - Circuits and Routing

Close

Overview | ExpressRoute | Virtual Network

One of the most important steps in implementing ExpressRoute is the creation of an ExpressRoute Circuit within a subscription. This is followed by configuring routing (Private Peering and/or Microsoft Peering).




**ExpressRoute configuration overview:**

**1. ExpressRoute Circuit configuration:**

The diagram to the left demonstrates the creation of a circuit:

- Provider / peering location: The connectivity provider and location
- Bandwidth: Total available bandwidth for use with the circuit
- SKU: Standard or Premium add-on (the Basic SKU is deprecated)
  - Premium provides additional functionality and limits
  - Refer to the ExpressRoute FAQ page for current information
- Billing model: Unlimited Data or Metered Data plans
  - Refer to the ExpressRoute Pricing page for current information

**2. Service provider provisioning and routing configuration**

- Provide the Service Key to the connectivity provider.
- Check the provisioning status for:
  - NotProvisioned - At the creation of the circuit
  - Provisioning - Service provider is provisioning the circuit
  - Provisioned - The circuit is provisioned and ready for use

ExpressRoute service providers can manage either **Layer 2 or Layer 3 circuits**. With Layer 3 circuits, you will work with your service provider to configure routing (Private Peering / Microsoft Peering).

**Key information for routing configuration is as follows:**

- Private peering requires private or public IP addresses for the routing interfaces (either 1 x /30, or 2 x /29 subnet).
- Public peering requires public IP addressing which you own:
  - Either 1 x /30 or 2 x /29 subnet
  - Microsoft must be able to verify the ownership of your IPs.
  - BGP communities are used to advertise Microsoft services.
  - More information can be found in this Microsoft article

# ExpressRoute - Virtual Network Connectivity

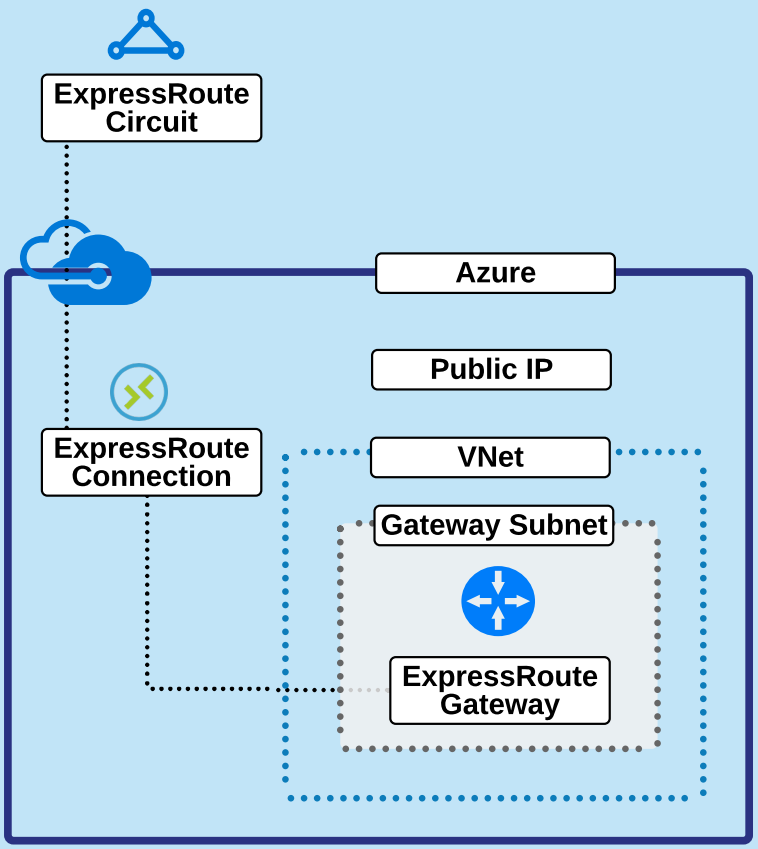
Close

Overview | ExpressRoute | Virtual Network

An ExpressRoute Circuit provides the secure connection between Azure and your connectivity provider. Associating that connectivity with your Virtual Network (VNet) requires a VNet Gateway.

**Important information about VNet connectivity with ExpressRoute:**

- This type of connectivity relies on the ExpressRoute Private Peering routing configuration.
- The VNet Gateway must be configured with ***type = expressroute***.
- The VPN-type of the VPN gateway must be route-based (dynamic).




**Configuration overview:**

The following independent resources must be configured and associated with the ExpressRoute VNet Gateway:

- Public IP: Address must be dynamic

The VNet Gateway must be configured as follows:

- Type: ExpressRoute
- Select SKU according to needs:
  - Basic (deprecated) does not support coexistence.
  - Standard/High/Ultra has network limits.
  - New zone-redundant gateways can be used.
  - Refer to this Microsoft article for current info.
- Must be associated to a public IP resource
- Must be deployed to a gateway subnet

The Connection resource is used to associate the ExpressRoute circuit with the VNet Gateway:

- Connection:
  - Type: ExpressRoute
  - ExpressRoute circuit: Associated circuit
- The Connection resource can be associated with circuits that exist outside of the current subscription.
- Multiple Connection resources ultimately allow multiple VNets to be associated with a single ExpressRoute.

# VPN Gateway - Overview

Close

Overview | Site-to-site | Point-to-site

VPN Gateways provide private and encrypted connectivity between an Azure Virtual Network (VNet) and one or more remote resources. Technically this is a VNet Gateway configured with **gateway type = vpn**.

**In this configuration, the VNet Gateway (often referred to as a VPN Gateway) provides support for:**

- Site-to-site (S2S) VPN: IPsec/IKE encrypted VPN tunnel with remote sites over the internet
- Point-to-site (P2S) VPN: IKEv2 or SSTP VPN with remote clients over the internet
- VNet-to-VNet: IPsec/IKE encrypted VPN tunnel with another VNet

**VPN Gateways have the following key configuration requirements:**

- VPN Gateways support the following SKUs:
  - Basic - A legacy SKU with a number of limitations
  - VpnGw1, VpnGw2, VpnGw3 - Differ based on connection limits and bandwidth
  - Inclusions regularly change, so for up-to-date information, see this Microsoft article
- VPN type:
  - Route-based (dynamic), which typically allows all traffic, and relies on route-tables to determine what networks are available for connecting, through the VPN
  - Policy-based (static), which includes network routes/prefixes being hard-coded in to the VPN connection itself
- All VPN Gateways must also be configured with the following related items:
  - Deployed to a **gateway subnet**, which should:
    - Be named "GatewaySubnet"
    - Only contain VNet Gateways
    - Have an address space sufficient to hold all VNet Gateways you will create
  - Linked to a **dynamic public IP address**

Refer to the Microsoft VPN Gateway FAQ for more details about VPN Gateways.

**Multi-Site VPN connections:**
Creating multiple concurrent S2S connections from one VPN Gateway to many different remote sites over the internet is possible. This is referred to as multi-site, and **requires** a route-based VPN type.

**A Note about VNet-to-VNet connections:**
VNet-to-VNet is very similar to S2S connections, with the main difference being that the Local Network Gateway address space is automatically created and populated.

# Site-to-Site (S2S) VPN

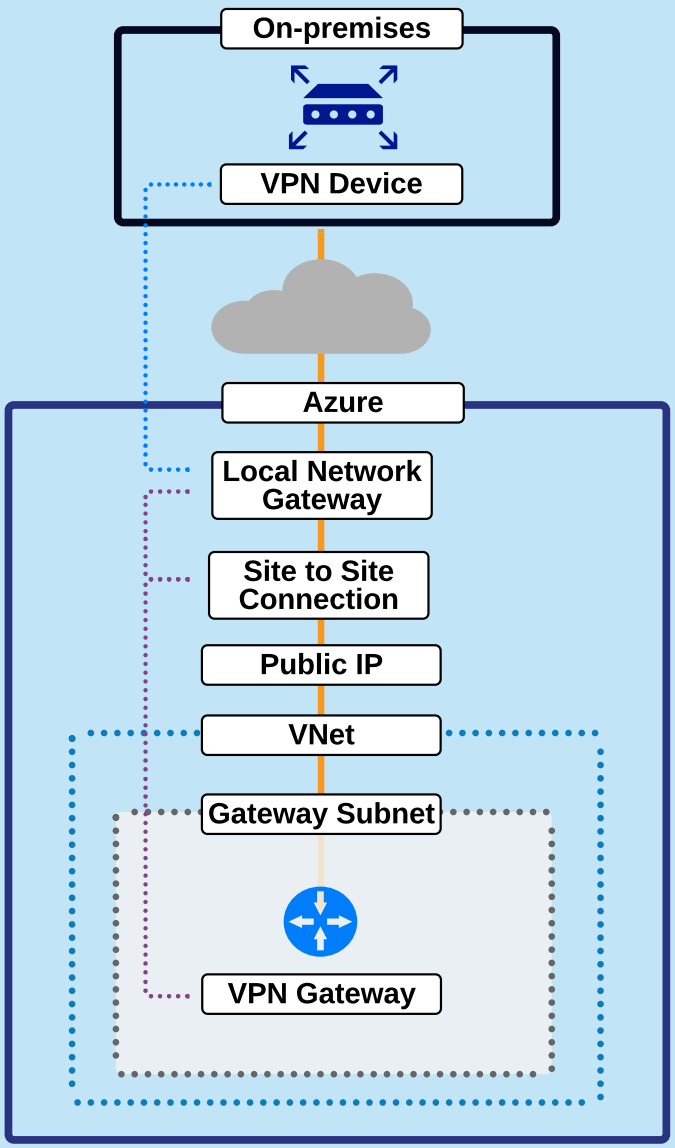
Close

Overview | Site-to-site | Point-to-site

Site-to-site (S2S) VPNs are configured to connect an Azure Virtual Network (VNet) with a remote site, over private IP addressing. This connection is encrypted with IPsec/IKE (IKEv1 or IKEv2) and relies on a VPN device located at the remote location.

**Important information about S2S connections:**

- They require a **VPN device** at the remote location with a **public IP, not** located behind NAT.
- Multiple S2S connections can be configured using one VPN Gateway with the RouteBased VPN type.




**Configuration overview:**

The following independent resources must be configured and associated with the VPN Gateway:

- Public IP: Address of VPN gateway (dynamic)
- Local Network Gateway (LNG): Defines the destination network
  - IP address of remote VPN device
  - Address space(s) available at remote site
  - BGP details if desired/required for remote site routing

**Configuring a VPN Gateway**

- Set up a VNet Gateway, including these main properties:
  - Configured with type=**vpn**
  - VPN-type: Either route or policy-based
  - SKU: Select any, according to needs
  - Must be associated to a public IP resource
  - Must be deployed to a gateway subnet

The Connection resource is used to establish the connection. This ties the VPN Gateway and the Local Network Gateway together:

- Connection: Establishes an encrypted tunnel
  - Connection type: S2S IPsec
  - Local Network Gateway: Associated to the LNG above
  - Shared key (PSK): Used for establishing the connection
- The connection resource includes downloadable VPN device configuration information

Close

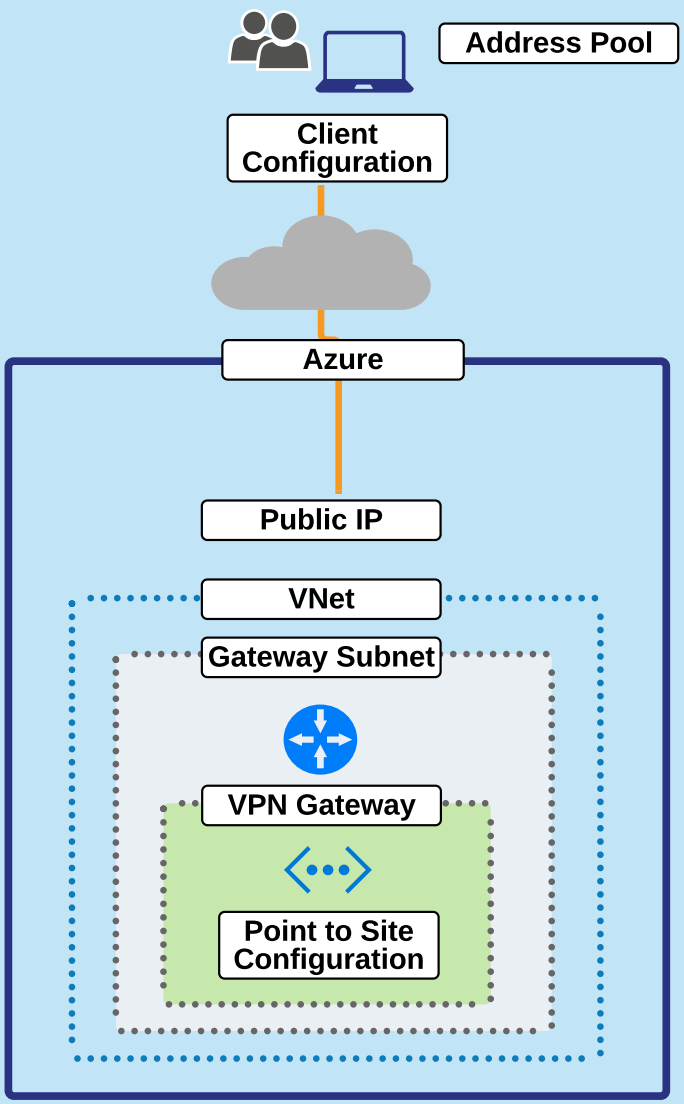
# Point-to-Site (P2S) VPN



Point-to-site (P2S) VPN allows a secure connection to be established between an individual client computer and an Azure Virtual Network (VNet). This supports Windows, Mac, and Linux.

**Important information about P2S configuration:**

- P2S supports a range of encryption, including IKE, IKEv2, OpenSSL, and SSTP.
- Client authentication can be configured through either certificates or RADIUS.
- The VPN-type of the VPN gateway must be route-based (dynamic).




**Configuration overview:**

The following independent resources must be configured and associated with the VPN Gateway:

- Public IP: Address of VPN gateway (dynamic)

**Configuring a VPN Gateway**

- Set up a VNet Gateway, including these main properties:
  - Type: **vpn**
  - VPN type: **route-based** (dynamic)
  - SKU: Any will work, according to needs
  - Must be associated to a public IP resource
  - Must be deployed to a gateway subnet
  - P2S Configuration (allows client computer connectivity)
    - Address pool: IP addressing to assign to clients
    - Tunnel type: for example SSL, IKEv2
    - Auth. type: Azure certificate or RADIUS

Key information about client configuration as follows:

- Depends on the authentication and tunnel type
- VPN client files can be downloaded from the portal

Close

# Virtual Network (VNet) Gateway

The Virtual Network (VNet) Gateway is the resource within Azure, created in order to establish private connectivity between external resources and an Azure VNet.

**There are two types of VNet Gateway:**

- **VPN** Gateway, which provides private connectivity with Azure over the public Internet:
  - VPN Gateways can be configured as a number of different connection types:
    - Site-to-site (S2S) VPN connections
    - Point-to-site (P2S) VPN connections
    - VNet-to-VNet connections
  - VPN Gateways support the encryption of traffic (IPsec/IKE encryption)
- **ExpressRoute** Gateway, which provides private connectivity with Azure:
  - Without using public Internet
  - Including higher security, reliability, and speeds (compared to VPN Gateways)
  - Relies on a ExpressRoute provider
  - Typically costs more than VPN connectivity
  - Does not include encryption

**Important information about VNet Gateways:**

- Both VPN and ExpressRoute Gateways must exist within a Gateway Subnet
- Site-to-site VPN and ExpressRoute can coexist, and requires:
  - Non-basic SKU gateways
  - Route-based VPN
- Each VNet can only have one VNet Gateway per *gateway type* (e.g. one ExpressRoute and one VPN)

# The Blueshift Guide

Subscription and Services Layer

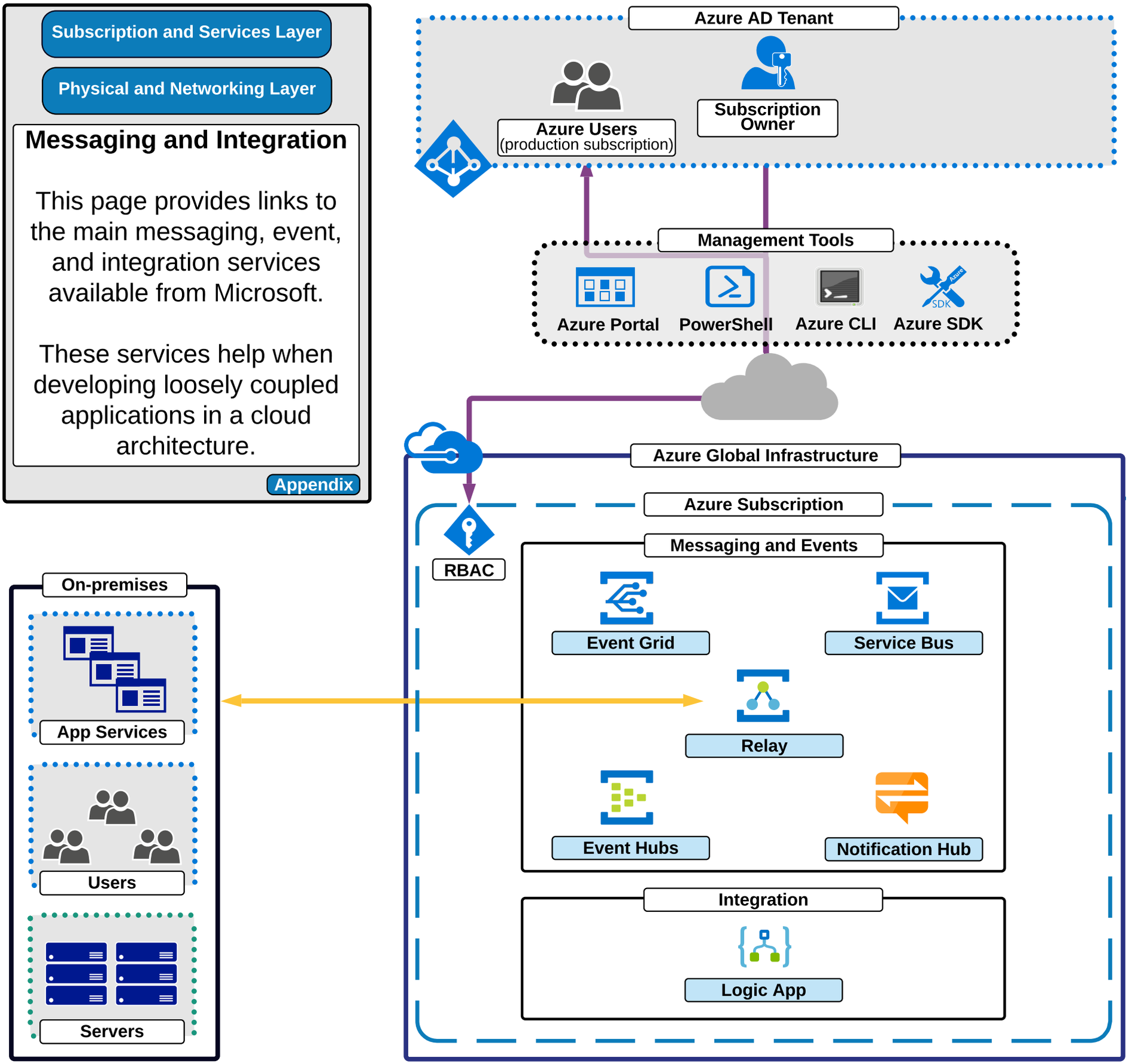
Physical and Networking Layer

## Messaging and Integration

This page provides links to the main messaging, event, and integration services available from Microsoft.

These services help when developing loosely coupled applications in a cloud architecture.

Appendix



Subscription and Services Layer

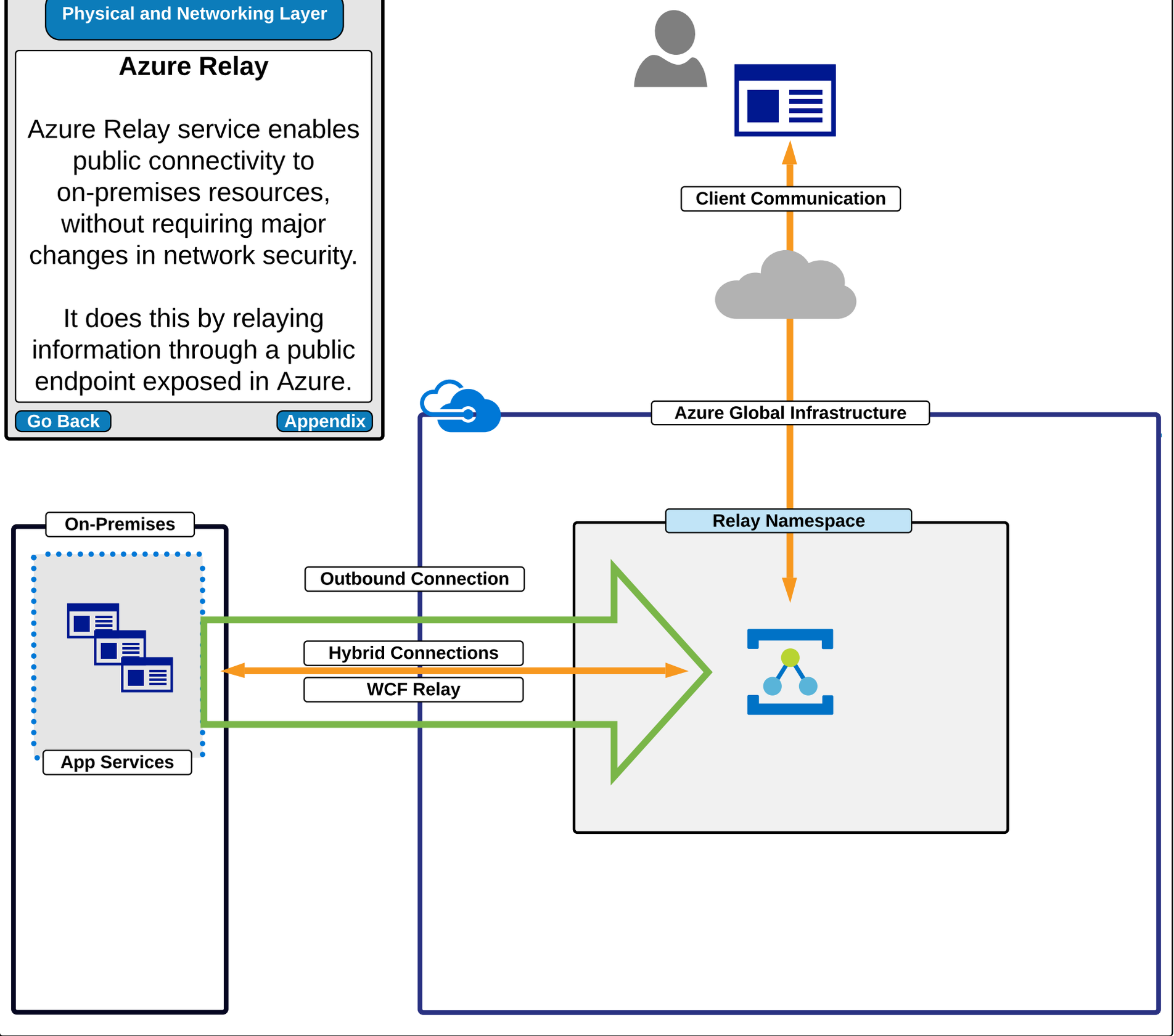
Physical and Networking Layer

## Azure Relay

Azure Relay service enables public connectivity to on-premises resources, without requiring major changes in network security.

It does this by relaying information through a public endpoint exposed in Azure.

Go Back Appendix



Subscription and Services Layer

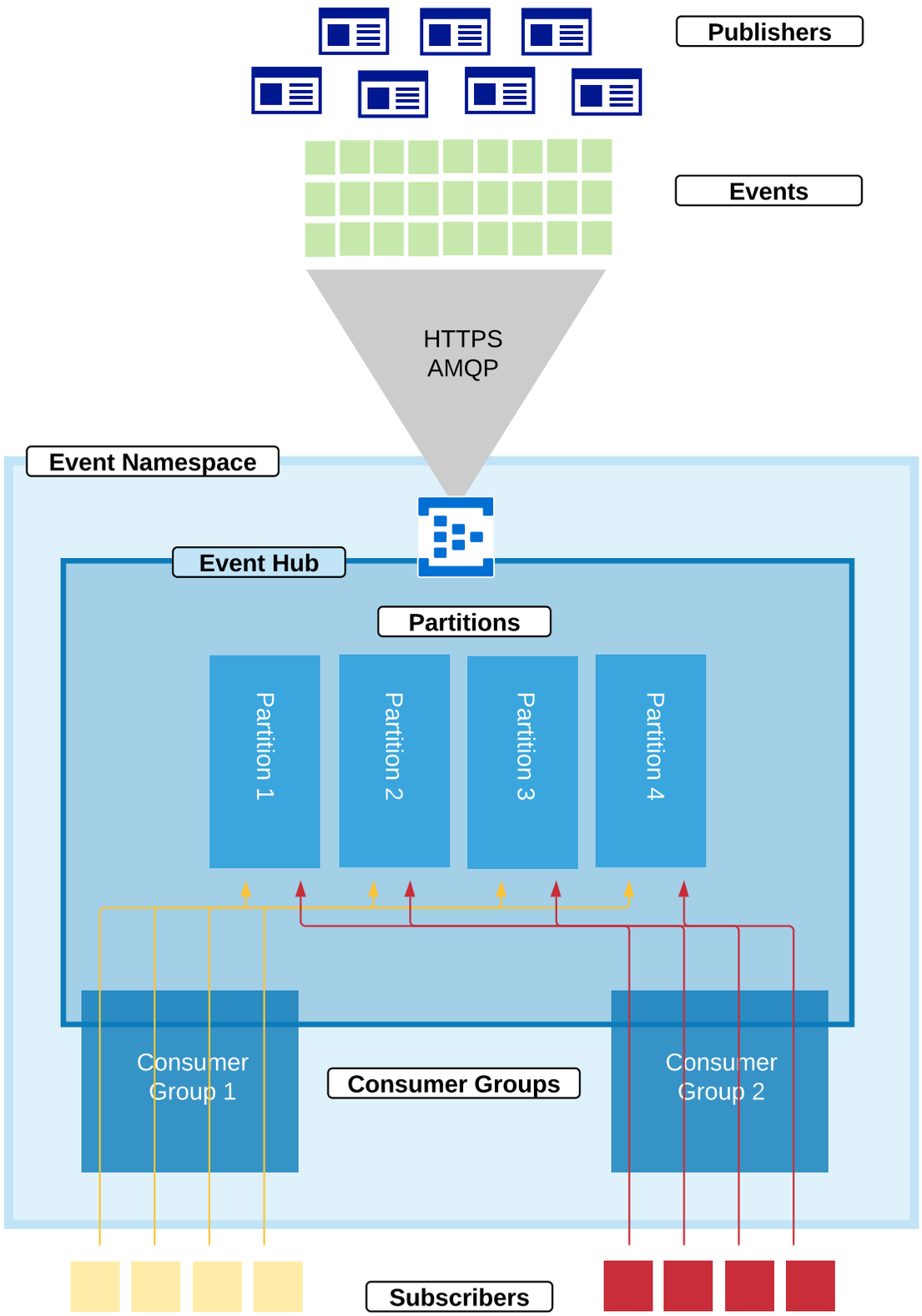
Physical and Networking Layer

## Azure Event Hubs

Azure Event Hubs is a highly scalable data streaming and event ingestion service.

It can be used to perform operations on large volumes of data. E.g. transaction processing, live dashboards, or anomaly detection.

Go Back Appendix



Subscription and Services Layer

Physical and Networking Layer

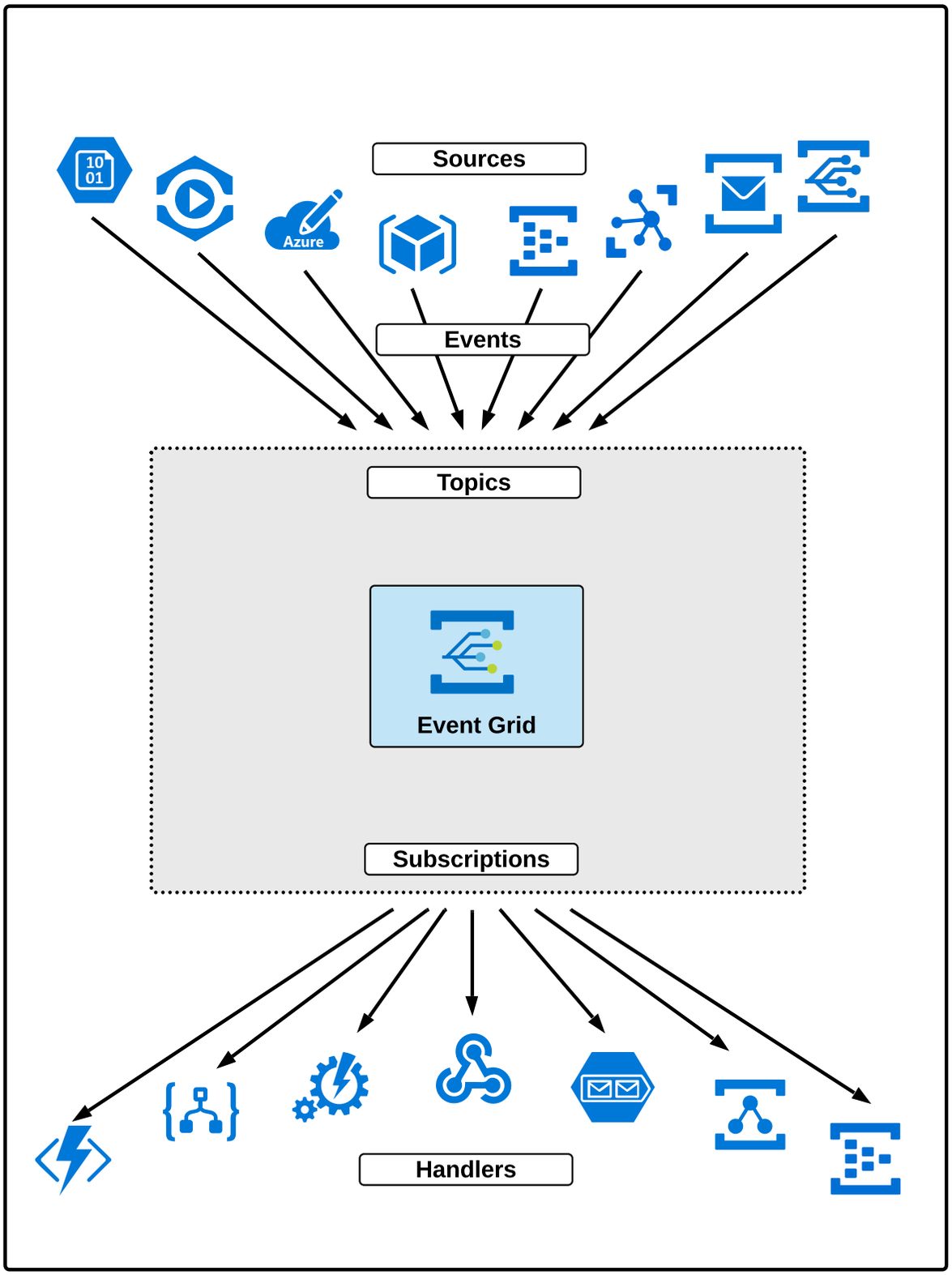
## Event Grid

As part of the messaging and event services available from Microsoft, Event Grid helps decouple applications and services.

Specifically, Event Grid provides a publish/subscribe managed service for the durable distribution of event information.

Go Back

Appendix



Subscription and Services Layer

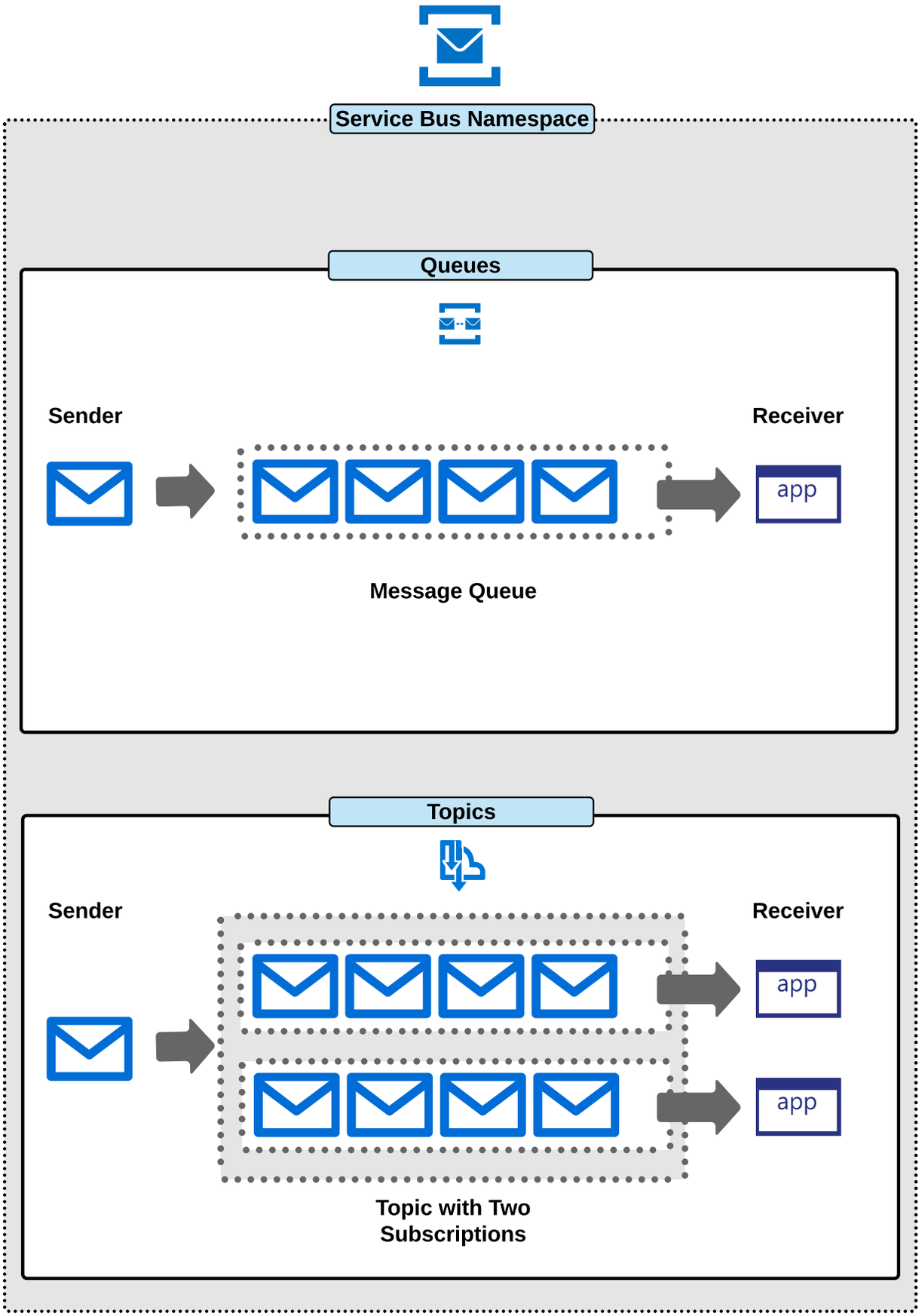
Physical and Networking Layer

## Service Bus

Service Bus is a brokered messaging service for traditional enterprise applications.

In contrast to other services, Service Bus should be used when handling high-value messages that cannot be lost or duplicated. Order processing and financial transactions are good examples.

Go Back Appendix



## Coming Soon

Close

The AZ-300 course is released **in preview**, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

To continue, please click the close (X) button.

# Event Grid

Close

Event Grid is a publish/subscribe solution for centrally managing the distribution of event information.

**Important information about Event Grid is as follows:**

- **Events** are small pieces of information which describe something occurring in a system/service:
  - The creation of a Virtual Machine is an event is a good example.
  - Events are limited to 64 KB of data.
- **Event sources** are where the actual events happen:
  - A range of Azure services are supported.
  - You can also create custom topics to support event distribution from custom applications.
- **Topics** represent the endpoint within Event Grid where Event Sources can send Event information to:
  - System topics are built-in and are not visible within subscriptions.
- **Event subscriptions** are relationships between a topics and event handlers:
  - This relationship defines what event information from a topic the handler is interested in.
  - Subscriptions can include event filtering (event type filtering, subject filtering, advanced filtering).
  - Subscriptions can be set to expire using Azure CLI.
  - Subscriptions include the endpoint for handling the event.
- **Event handlers** are the endpoint responsible for handling the event, based on the subscription.

**Important information about the delivery of event information is as follows:**

- Event Grid delivers each message at least once for each subscription:
  - Events are sent to the registered endpoint of each subscription immediately.
  - Events are sent individually to subscribers as an array including a single event.
- HTTP response codes are used to indicate success or failure of delivery:
  - Success codes include 200 OK and 202 Accepted.
  - Failure codes include 400, 401, 404, 408, 413, 414, 429, 500, 503, and 504.
- If Event Grid can't confirm an event has been received, it redelivers the event:
  - The retry attempts start after 10 seconds, and then use exponential backoff (30s, 1min, 5min, etc.).
  - Undelivered events expire after 24 hours, and can be sent to a storage account (dead-lettering).
- Event Grid authentication (WebHook event delivery, event subscriptions, custom topic publishing):
  - Refer to this Microsoft article for the various authentication methods available.

# Logic Apps

Close

Logic Apps are often referred to as "workflow-as-a-service." Using either graphical design or code, it is possible to to build workflows which integrate and control services in response to triggers.

Logic Apps integrate with a range of different services, such as the typical HTTP request, schedule, or Azure services, as well as third-party providers such as Twitter, Facebook, or Outlook.

**Steps to create a Logic App:**

- First, the resource group and actual Logic App resource must be created.
- Then, using either the designer or Code view, a **workflow** can be created.
- The workflow will likely access data, services, or systems, and can do so using **managed connectors**.
- Using data or event information from these connections can then **trigger** steps in the logic app to run.
- Based on a given trigger, an **action** can will be executed.

**Important configuration and management information about Logic Apps:**

- Logic App underlying code uses Workflow Definition Language in JavaScript Object Notation.
- Managed connectors support both triggers and actions for other services, and are summarized as:
  - **Managed API connectors**: Access Microsoft and other third-party services (e.g. Twitter)
  - **On-premises connectors**: Access on-premises resources with the *on-premises data gateway*
  - **Integration account connectors**: Connect, transform, and validate business-to-business messages
  - **Enterprise connectors**: Access systems such as SAP and IBM MQ
- Built-in actions and triggers are available such as scheduling, nested logic, and data manipulation
- Logic Apps support additional security through a number of means, for example:
  - Incoming IP addresses allowed to access the Logic App can be restricted.
  - Logic App access to other services can be controlled through Azure AD Managed Identities.
  - URLs used to access Logic Apps are protected with SAS, encrypted by the Access Key.

# Azure Relay

Close

Azure Relay service provides a secure method for exposing on-premises applications over the public Internet, by allowing communication to be relayed via Azure to on-premises entities.

**An overview of the flow of data when using Azure Relay is as follows:**

1. An on-premises service establishes an outbound connection to the Azure Relay service.
   *Note:* This is why only outbound connections are required although we are providing public access)
2. The outbound connection establishes a bi-directional socket
3. Clients can communicate with the on-premises service publicly, by communicating with the publicly accessible Azure Relay namespace address
4. The Azure Relay service then utilizes the bi-directional socket to allow communication.

**A summary of important information about Azure Relay:**

- Using Azure Relay to provide public accessibility to on-premises applications:
  - Does not require inbound firewall rules on-premises
  - Only requires an outbound connection between on-premises and Azure Relay
  - Supports bi-directional communication
  - Does not require on-premises hardware
  - Can be scoped to individual applications/services
- Authentication and authorization is supported using:
  - Shared Access Signature authentication, for managing rights

**Azure Relay supports both Hybrid Connections and WCF Relays, as summarized below:**

- **Hybrid Connections**: Supports open standard web socket communication
- **WCF Relays**: Only supports Windows Communication Foundation (WCF) communication using remote procedure calls (RPC)

Close

# Notification Hubs

Notification Hubs is a fully managed, scalable solution for managing push notifications to various platforms (iOS, Android, Google, Windows, Kindle, Baidu, etc.).

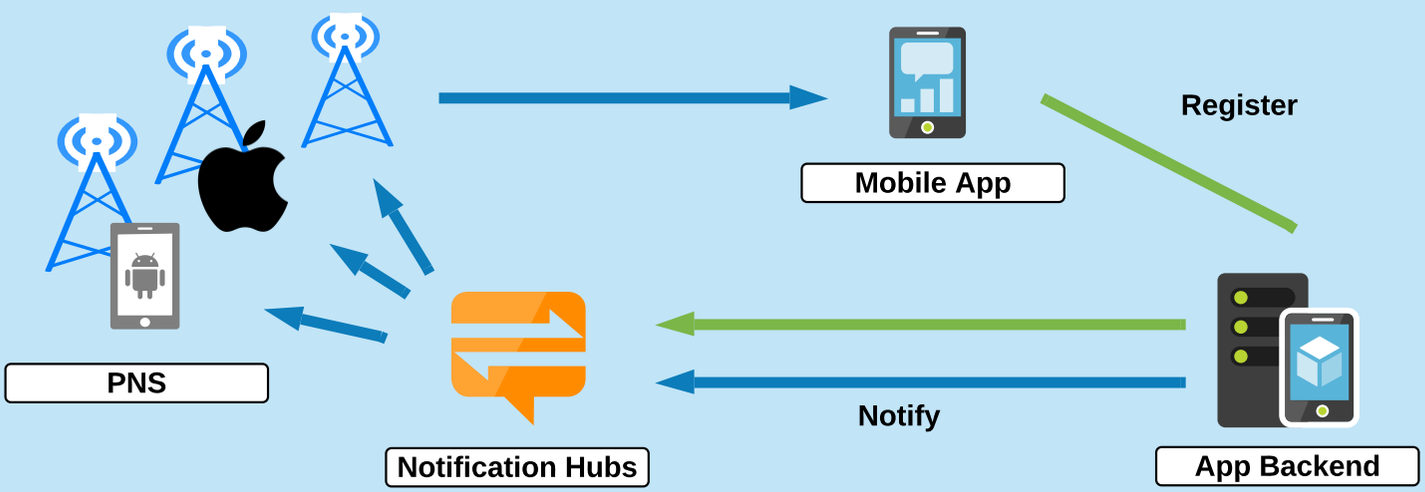
**An overview of the typical setup of push notifications using Notification Hubs is as follows:**

- Push Notification System (PNS) registration is typically required by each vendor/platform.
- Notification Hub is created & configured for the given vendor/platform push notifications.
- An application is registered/connected to the Notification Hub.
- Push notifications can then be submitted to registered applications/devices.

**A summary of Notification Hub configuration is as follows:**

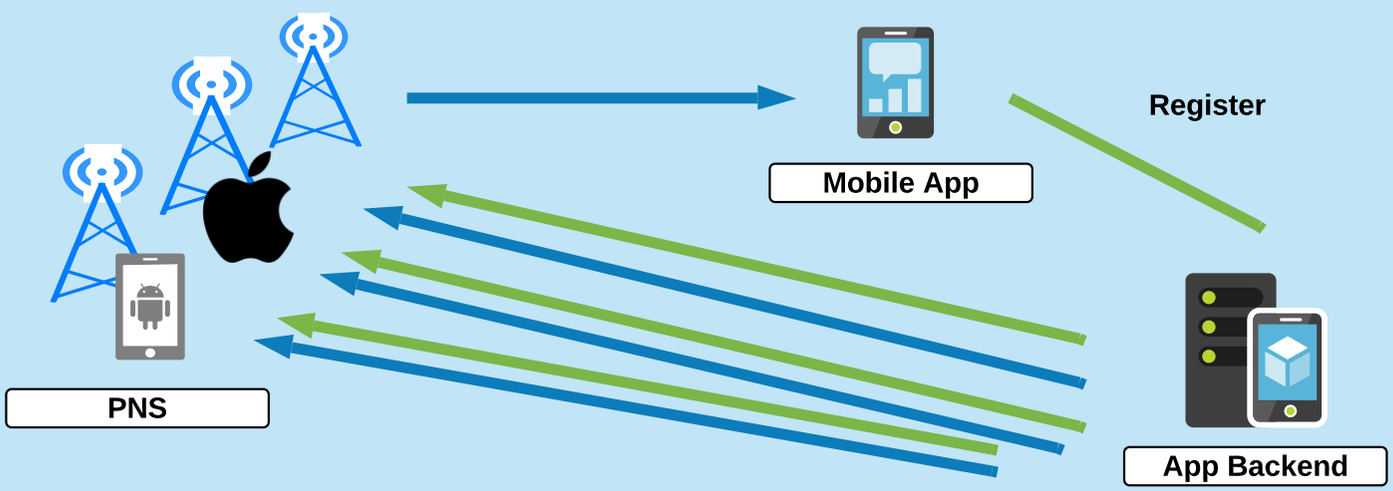
- A **namespace** must first be configured, which includes:
  - **Name:** Unique name of the namespace,
  - **Pricing tier**, which currently includes Free, Basic and Standard:
    - Pricing determines notification limits, monthly cost, features, and more.
    - For more information on the different limits and features, refer to this Microsoft pricing link.
- Within a namespace, a **hub** can be configured, as follows:
  - Typically a single hub is created for an individual application.
  - Configuration is required for each vendor you wish to provide push notifications for, including:
    - Some sort of registration (dependent on the vendor) typically including key and authentication
- Authentication is configured using Shared Access Signature Security (SAS):
  - Two rules are automatically created with the Notification Hub:
    - SAS rule with *listen* rights (for client app registration)
    - SAS rule with *all* rights (for the application backend)

Close




Toggle

Close




Toggle

# Event Hubs

Close

Overview | Namespace | Event Hub

Azure Event Hubs provides a massively scalable solution capable of processing millions of events each second. A common example of Event Hubs is fraud detection for major banks. In this scenario Event Hubs can be used to ingest and process large volumes of transactions.

To get started creating an Event Hub, we must first create an Event Hubs namespace.

**Important elements of an Event Hubs solution:**

- **Events:** Small pieces of information about something that has happened:
  - Often referred to as a datagram
  - Can be published individually or in batches
  - A single publication of event(s) cannot exceed 256 KB
- **Publishers:** An application, service, or device which emits an event:
  - Publishers send event data using HTTPS or Advanced Message Queuing Protocol (AMQP) 1.0.
- **Subscribers:** Applications which receive data from an Event Hub using one of two methods:
  - **Event Processor Host:** Simplified higher-level method receiver (depends on Event Hub Receiver)
  - **Event Hub Receiver:** Lower-level method with greater complexity

# Event Hubs Namespace

Close

Overview | Namespace | Event Hub

A namespace is the parent container of any Event Hub you wish to create. You can create several Event Hubs within an Event Hubs namespace. Settings at the namespace layer will apply to all child Event Hubs.

**Important information about the Event Hubs namespace:**

- Several important properties are configured at the namespace level, including:
  - The unique name, which also creates the DNS entry
  - The pricing tier which, currently includes Basic, Standard and Dedicated:
    - Pricing determines connection and consumer group limits, and throughput.
    - For more information on the different limits and features, refer to this Microsoft pricing link.
  - Whether any zone redundancy is required
  - Whether an Apache Kafka endpoint should be available
  - Auto-Inflate functionality which will automatically scale-up the number of throughput units
- Authentication and authorization through shared access signatures (SAS) applies at this level.

# Event Hub



An Event Hub is created within an Event Hub's namespace, and is where all events are published and read from. An Event Hub can be configured with a number of features which improve scalability and performance.

**Important information about Event Hub components and properties is as follows:**

- An Event Hub is configured with these attributes:
  - **Name** of the Event Hub (which must be unique within the subscription)
  - **Partition Count:** The number of partitions for your Event Hub:
    - Must be between 2 and 32
    - Is typically configured to match the planned number of concurrent consumers
    - Cannot be changed after creation
    - Enables the "partitioned consumer pattern"
  - **Message Retention:** The number of days messages will be retained for (between 1 and 7):
    - Data stored in Event Hubs will exist until the Message Retention period expires.
- **Consumer groups:** A view of the data within the Event Hub:
  - Data exists within an Event Hub event
    *Note:* it is also possible to define SAS authentication for individual consumer groups
- **Capture:** It is possible to stream data to Azure Blob storage or an Azure Data Lake Store account.

# Service Bus

Close

Overview | Queues | Topics

Service Bus is a messaging broker designed to deliver messages between decoupled applications in a highly-available, highly-reliable way. Service Bus supports the "competing consumer" pattern.

To get started creating a Service Bus, we first must create a namespace.

**Important information about the Service Bus Namespace is as follows:**

- A namespace is the parent container of any Service Bus Queues or Topics you wish to create.
- Several important properties are configured at the Namespace level, including:
  - The **name**, which also creates the DNS entry name.servicebus.windows.net
  - The **pricing tier** which currently includes Basic, Standard and Premium:
    - Only Standard or Premium currently support Topics and Subscriptions.
    - For more information on the different limits and features, refer to this Microsoft pricing link.
  - Whether any zone redundancy is required (currently only supported with Premium pricing tier)
- Authentication and authorization through shared access signatures (SAS) applies at this level.

# Service Bus - Queues

Close

Overview | Queues | Topics

Service Bus Queues are the standard messaging entity, available in all pricing tiers. Queues provide **First In, First Out (FIFO)** message delivery to one or more consumers.

**Important information about Queues:**

- **First In, First Out (FIFO)** message delivery, which supports two different receive modes:
  - ReceiveAndDelete
    - Least guaranteed form of message delivery
    - Immediately marks a message as having been consumed upon receive request
    - A message will be marked as consumed even if the consumer fails to fully process it
  - PeekLock
    - Provides the greatest reliability for applications which cannot tolerate missing messages
    - Two-step receive process which locks a message until the consumer confirms the message has been fully received (CompleteAsync), or if receive is abandoned (AbandonAsync)
- **Measage TTL:** Sets the default lifetime of a message, after which time the message will expire
- **Lock Duration**: Default duration of a lock during PeekLock receipt of the message
- **Duplicate detection:** Whether or not Service Bus will detect and delete duplicate messages
- **Dead-lettering:** Supports undelivered messages being saved to a special dead-letter-queue (DLQ)
- **Sessions:** Allow for the use of a SessionID to help manage sequences of related messages
- **Partitioning:** Provides transparent partitioning across multiple message brokers

# Service Bus - Topics and Subscriptions



Topics and Subscriptions are a useful feature of Azure Service Bus, helping with scaling to a large number of message recipients. This differs from queues where each message is processed by a single consumer.

**Important information about Topics and Subscriptions:**

- First, it helps to understand the architecture of topics and subscriptions:
  - A topic is where a sender submits its message.
  - A subscription is like a virtual queue that messages can be received from.
  - Messages must be submitted to a topic, and received from a queue.
- A subscription can use filters to determine which messages from the topic will be available:
  - **TrueFilter** selects all messages.
  - **FalseFilter** selects none of the messages.
  - **SqlFilter** uses an SQL-like expression to determine which messages are available.
  - **CorrelationFilter** evaluates user-defined and system properties for suitable messages.

Subscription and Services Layer

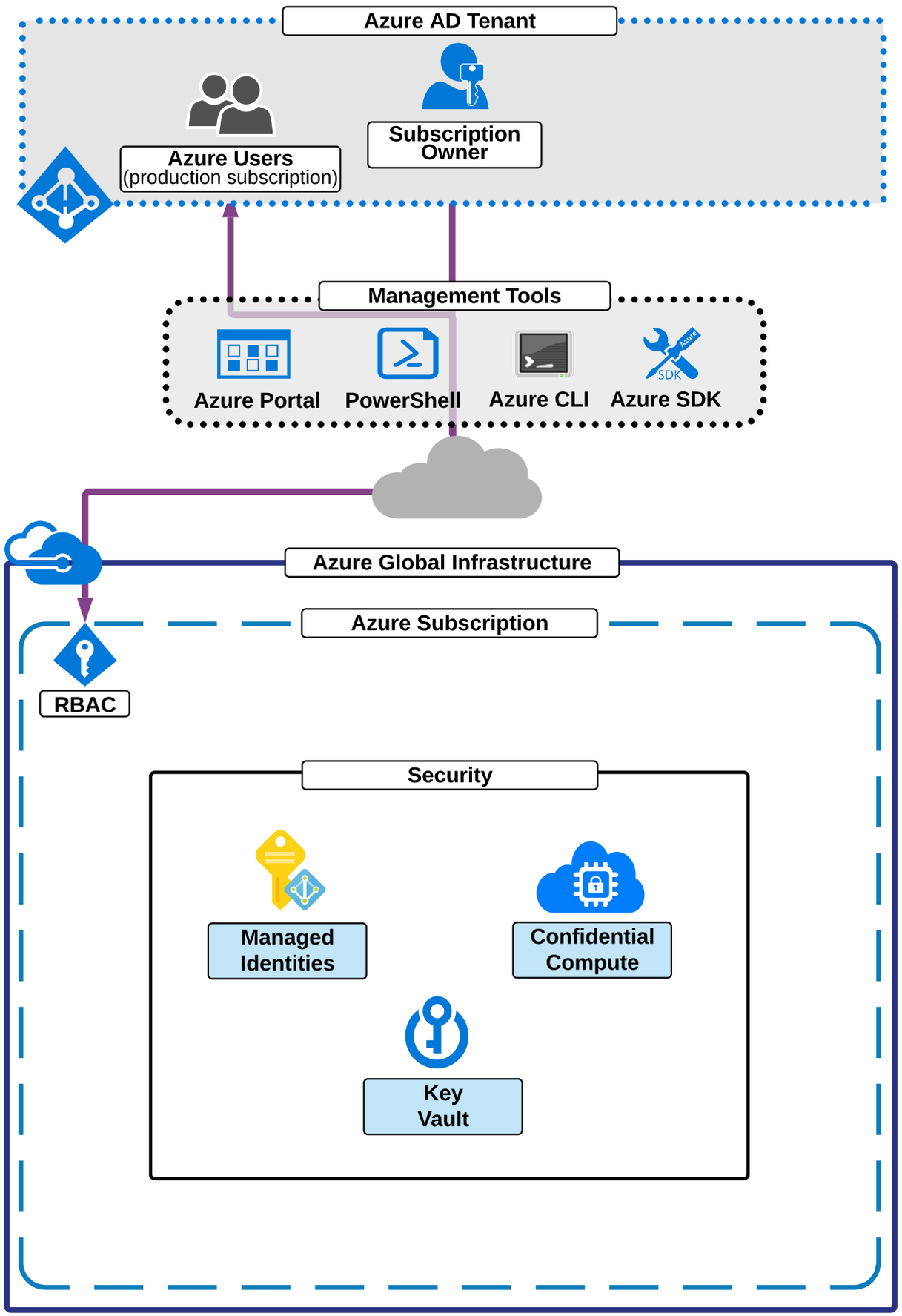
Physical and Networking Layer

## Security

Azure provides a number of different services to protect your resources within Azure.

This page lists a number of the main services you should be familiar with for AZ-300.

Appendix



## Coming Soon

Close

The AZ-300 course is released **in preview**, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

To continue, please click the close (X) button.

Close

# Managed Identities

Managed Identities provides a secure method for authenticating Azure resources against other Azure services, without needing to include credentials. Managed Identities is a feature of Azure AD which specifically provides an Azure resource with a managed identity within Azure AD.

This feature provides the ability to authenticate an Azure resource "behind-the-scenes." This does not provide any implicit permissions (authorization) though. That must still be configured separately.

**Important facts about Managed Identities:**

- Avoids the need for application credentials to be stored in code (e.g. Client ID and secrets)
- Is fully managed by Microsoft, so credentials no longer need to be rotated by developers
- Automates the creation/registration of an application within Azure AD, Service Principal, and Client ID
- Includes built-in functionality for Azure resources to securely obtain an authentication token
- Does not imply any authorization, since the identity must still be granted whatever permissions are desired

**Important components of Managed Identities:**

- Managed Identities include three main elements:
  - **Principal ID:** The object ID of the service principal object for the managed identity
  - **Client ID:** Unique identifier generated by Azure AD
  - **Azure Instance Metadata Service (IMDS):** REST endpoint accessible to VMs via 169.254.169.254
- Two types of managed identities are available:
  - System-assigned:
    - Fully managed by Azure and directly created for and associated with a resource
    - Exists with the resources, and is deleted if the resource is deleted
    - Associated with a single resource
  - User-assigned:
    - Created as an individual resource, managed by an administrator (e.g. you)
    - Must be managed (created, deleted, etc.) by an administrator
    - Can be associated with one or more Azure services

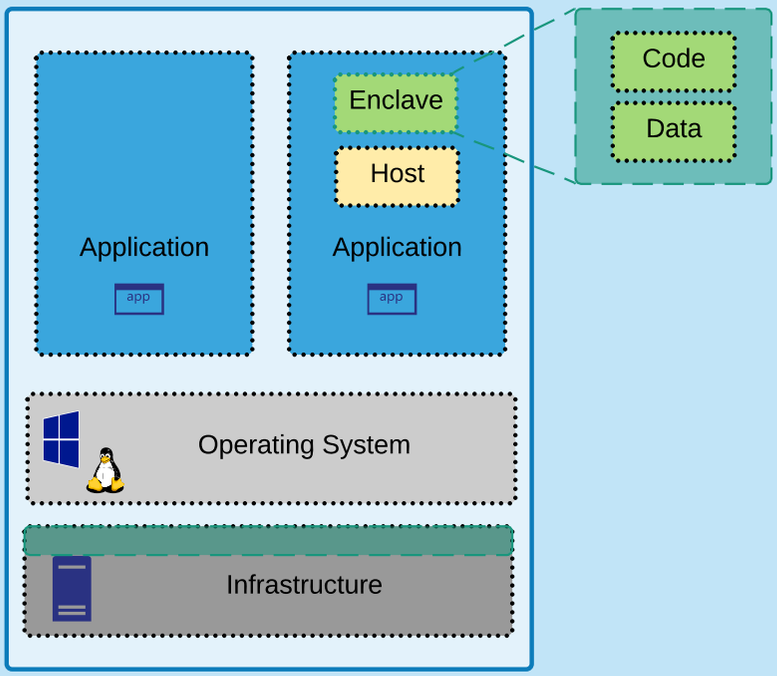
# Confidential Compute

Close

Confidential Compute refers to a number of efforts in research, hardware, and software, which are aimed at protecting data whilst it is being processed in the cloud.

This represents a focus on encryption/security of data "in use." It differs from the common models of encryption at rest or encryption in transit, and is primarily comprised of the following:

- Security is achieved through the use of a protected **Trusted Execution Environment (TEE):**
  - TEEs are also referred to as *enclaves*.
  - Enclaves provide a highly secure, isolated area for code and data.
  - Applications can be developed to use these enclaves, using the **Open Enclave SDK**.
- Microsoft Azure supports the use of TEE via:
  - Compute-based, using Intel's Software Guard eXtensions (SGX) which is available in the DC series VMs
  - An expanding number of services which support the Open Enclave SDK




**Information about the use of a TEE:**

- It is not possible to view data/operations within a TEE from outside of the TEE
- Sometimes this is referred to as trusted/untrusted
- More information on Open Enclave SDK can be found at this Microsoft Github page

# Key Vault

Close

Key Vault is a service which allows for secure storage, and retrieval, of cryptographic keys and other secrets. It is programmatically accessible, and stores secrets in hardware security modules (HSM).

**High level summary of Key Vault features is as follows:**

- Allows you to store a range of items, including passwords, tokens, certificates, API keys, and more
- Can be used for key management, supporting the creation, control, and rotation of encryption keys:
  - Keys can be imported/generated in HSMs certified to FIPS 140-2 level 2 standards
- Enables certificate management, including enrollment of certificates from third-party Public CAs

**Important Key Vault properties include:**

- Name of the Key Vault, which creates a Vault URI: https://name.vault.azure.net
- Access Policies, which can be used to control access on the data-plane of the Key Vault
- Pricing tier including standard or premium (only premium supports HSM-protected keys)

**An overview of Key Vault management through the REST API:**

- Key Vault can be managed and operated through the Key Vault REST API.
- This includes operations such as management, creation, and deletion of a Key Vault or keys/secrets/certs.
- All requests to Azure Key Vault MUST be authenticated:
  - Key Vault supports Azure AD access tokens using OAuth2.
  - Managed Identities can be used to achieve authentication.
- Refer to this Microsoft Article for more information on the various REST API commands.
- An example REST API call, for creating/updating a Vault is as follows:

  **HTTP GET**

  ```
  https://management.azure.com/subscriptions/{subscriptionId}/resources?$filter=resourceType eq 'Microsoft.KeyVault/vaults'&api-version=2018-02-14
  ```

Subscription and Services Layer

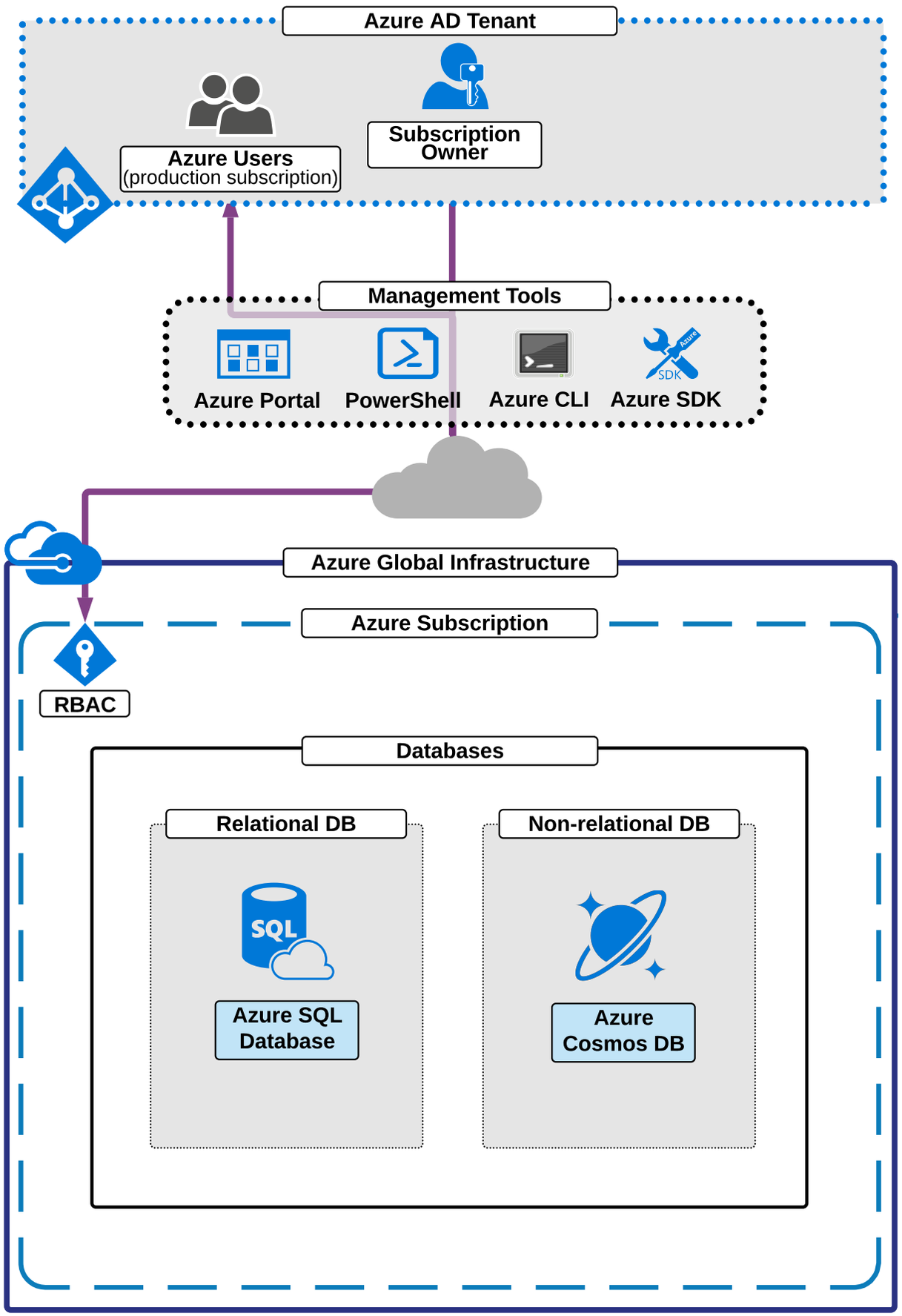
Physical and Networking Layer

## Databases

Microsoft provide a range of data storage and processing services.

The two main data database facilities are Azure SQL and Azure Cosmos DB.

Appendix



## Coming Soon

Close

The AZ-300 course is released **in preview**, and more content will be coming soon. Please check back later.

If you believe this is an error, then please reach out via community, or lodge a ticket for support.

To continue, please click the close (X) button.

# Azure SQL Database

Close

Overview | Encryption

Azure SQL Database is a database-as-a-service solution fully managed by Microsoft. It provides features and functionality similar to traditional Microsoft SQL Server, for relational databases.

**Important information about Azure SQL Database:**

- Created with with the following settings:
  - SQL Logical Server, which configures: and a location
  - Pricing tier which sets resource and storage limits
- Supports a range of features like firewall, geo-replication, backups, etc.

**Azure SQL Database pricing models:**

- Database Transaction Units (DTU) - An abstracted representation of the underlying resources
- vCPU - A clearer view of actual underlying resources, with control over storage and CPU

**Information about Azure SQL Database deployment options:**

- Single database: Managed SQL Database server, recommended for cloud-born applications
- Elastic pools: A resource pool intended to be shared by multiple single databases
- Managed instance: Intended for databases migrated from on-premises with near 100% compatibility with on-premises SQL server

**Elastic Pools:**

Elastic Pools can improve and simplify the manageability of scale when multiple databases are involved. Pools are suitable when multiple databases have low average utilization and relatively infrequent utilization spikes.

# Azure SQL Databases - Encryption

Close

Overview | Encryption

Microsoft provides two main methods for managing encryption on Azure SQL Database - Transparent Data Encryption (TDE) and Always Encrypted.

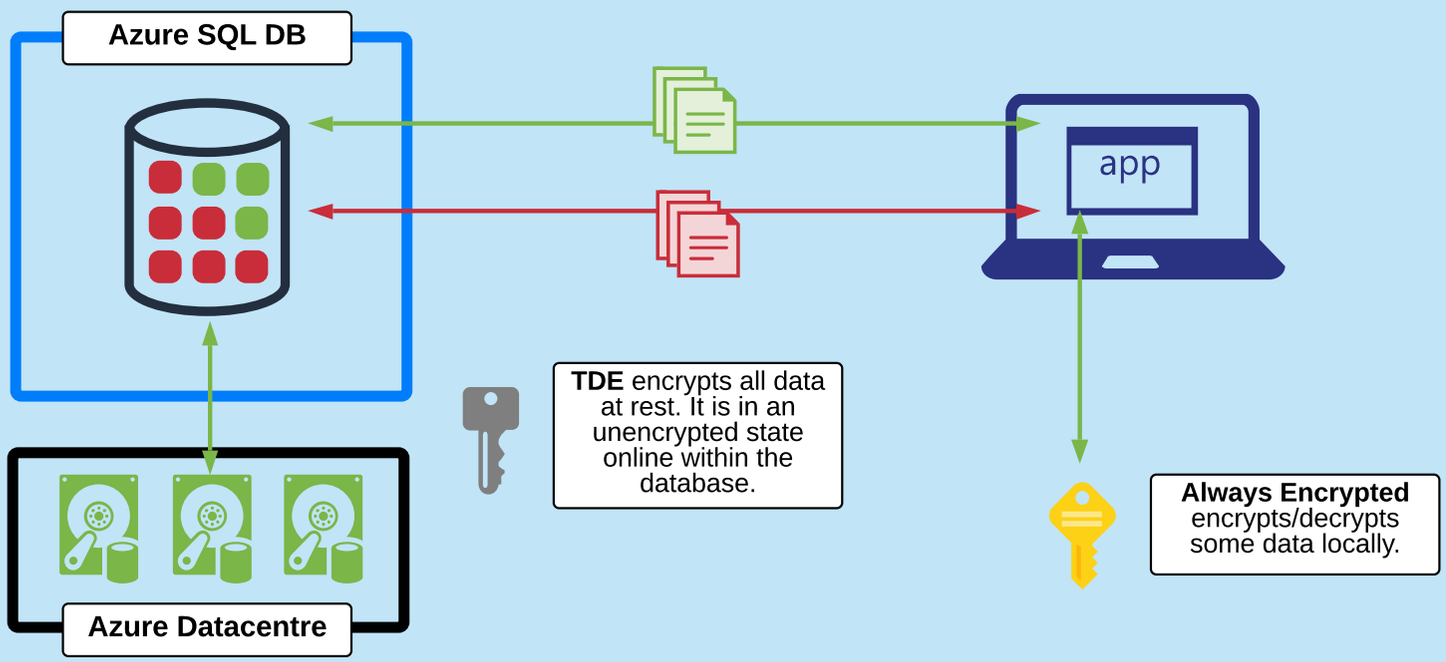
The following is a list of the key features and functions of both methods:

**Transparent Data Encryption (TDE):**

- The focus of TDE is protection of data at rest - e.g. datacentre of storage theft
- TDE performs decrypt/encrypt on the underlying server, transparent to the application
- Information within the database is accessible/visible to any who have access to the data (unlike below)

**Always Encrypted:**

- The focus of Always Encrypted is protection of confidential/sensitive information from DB administrators
- The flow of traffic for always encrypted is as follows:
  - Client computers have an Always Encrypted driver installed,
  - Confidential/sensitive data (e.g. credit card information) is encrypted client-side,
  - The encrypted data is transferred to the database in encrypted form,
  - The encrypted data is stored in the database in encrypted form.
- Client applications never reveal the encryption keys to Azure SQL



# Cosmos DB

Close

Overview | Partitioning | Consistency

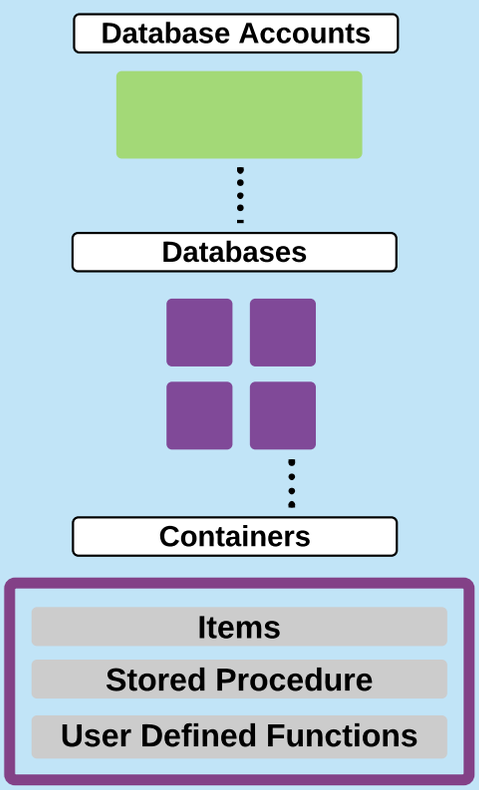
Cosmos DB is an advanced database service designed for global data distribution and accessibility. Microsoft manages the underlying infrastructure to provide scale, availability, performance, and replication for a range of non-relational data types.

**Some of the main Cosmos DB features include:**

- Multi-master geographic distribution and transparent replication of data across Azure regions
- Five well defined consistency levels for replication spanning strong <> eventual
- Elastic horizontal scalability and single-digit-millisecond accessibility
- Support for multiple data types and APIs, including SQL, MongoDB, Cassandra, and Gremlin
- Server-side programming including stored procedures, triggers, and user defined functions

**Important information about Request Units which are used for managing throughput:**

- 1 RU is calculated as the throughput required to read a 1 KB document
- Abstracts the capacity planning difficulties of understanding required CPU/memory/IOPS
- Can be provisioned at either the Database or Container layer




**Cosmos DB is made up of the following components:**

- Cosmos DB Account includes the following properties:
  - Name: establishes the URI: name.documents.azure.com,
  - API: for data type / API,
  - Location: initial location (can be expanded with replication).
- Database - namespace / management layer for containers
- Containers - unit of scalability:
  - Type of container depends on data type / API,
  - Can be Collections (document oriented APIs), Graphs (Gremlin API), Tables (table API), etc,
- Items - actual entities being stored
  - Based on  type (data type / API)
  - Can be Documents (document APIs), Tables (Cassandra), Graph (Gremlin), etc.

# Cosmos DB - Partitioning

Close



Partitioning is used at the Container layer so that the Cosmos DB service can distribute data amongst infrastructure so as to maintain performance levels.

You must specify a partition key for the Cosmos DB service to use to evenly distribute data and access.

**Important information about partitioning:**

- Partitioning ***divides items in a container in to logical partitions***, based on the partition key
- Items within a container will be sent to logical partitions based on the hash of partition key of the item
- Partitioning is important and can impact:
  - Pricing - queries that access data in a single partition are cheaper than across multiple partitions
  - Transactions in stored procedures or triggers can only be performed against a single partition
- Partition management, scalability, and distribution

**Considerations for selecting a partition key:**

- Each partition has a limit of 10 GB of storage; partitioning should consider your ***storage*** requirements
- Throughput is limited at the partition layer; partitioning needs to consider read/write ***request***-distribution (e.g. partitioning based on timestamp would result in a hotspot for requests)
- Transactions can be scoped to a partition key; consider your top queries & common scope/filters

# Cosmos DB - Consistency

Close



A major consideration of any distributed database is balancing the levels of availability, performance, and consistency. Most distributed databases have two options: strong constency, and eventual consistency.

With Cosmos DB, there is much greater control over this balance, as depicted below.

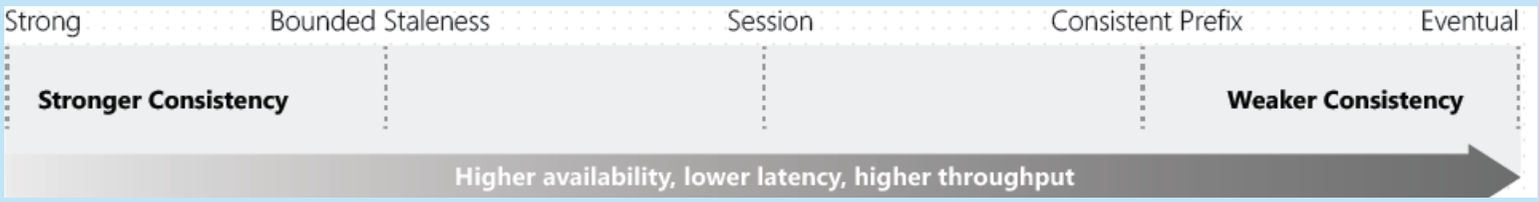



*Image taken from this Microsoft article on Cosmos DB Consistency Levels.*

**Important information on Cosmos DB consistency levels:**

- The five consistency models from strongest to weakest are: strong, bounded staleness, session, consistent prefix, and eventual
- Consistency levels are region agnostic and are guaranteed for all read operations
- Default consistency levels are configured at the Cosmos DB Account level
- Consistency levels can be overriden through the SDK for a whole client or per-request

**Information for each consistency level:**

- **Strong:** reads are guaranteed to return the most recent committed version of an item.
- **Bounded staleness:** will be consistent to an agreed amount, this can be configured in two ways:
  - Reads might lag behind writes by at most ___ version of an item, or
  - Reads might lag behind writes by at most ___ amount of time.
- **Session:** reads are guarnteed to honor writes for a given client session.
- **Consistent prefix:** guarantees that reads never see out-of-order writes.
- **Eventual:** no guarantee about orders of reads; replicas will *eventually* converge.

Subscription and Services Layer

Physical and Networking Layer

## Appendix

Useful links and information for the AZ-300 exam.

Appendix

# Helpful Links

About the Exam | Helpful Links

### Helpful links:

- AZ-300 Exam description and requirements: https://www.microsoft.com/en-us/learning/exam-az-300.aspx
- Microsoft Certified Azure Solutions Architect Expert: https://www.microsoft.com/en-us/learning/azure-solutions-architect.aspx
- Microsoft Learning Blog: https://www.microsoft.com/en-us/learning/community-blog.aspx
- Sign up for a free Azure subscription: https://azure.microsoft.com/Account/Free
- Linux Academy blog: https://linuxacademy.com/blog/
- Instructor contacts (James Lee):
  - LinkedIn - https://www.linkedin.com/in/james-lee-6551a314/
  - Twitter - https://twitter.com/jamesdplee
  - Email - james.lee@linuxacademy.com

Subscription and Services Layer

Physical and Networking Layer

Appendix

Useful links and information for the AZ-300 exam.

Appendix

# About the Exam

About the Exam | Helpful Links

**AZ-300 Exam:**

- **Exam Length:** 150 minutes
- **Number of Questions:** 40 to 60 questions
- **Exam Question Format:**
  - Active screen
  - Best answer
  - Build list
  - Case studies
  - Drag and drop
  - Hot area
  - Multiple choice
  - See this Microsoft article for information on question types
- **Exam Cost:** $165.00 USD
- **Exam Registration:** https://www.microsoft.com/en-us/learning/exam-az-300.aspx

Note: the AZ-300 exam can be taken at a test centre, or as an online-proctored exam (performed online). There are special conditions for online-proctored exams. See this link for more info.

***Preparing for the AZ-300 Exam:***

- Attempt all hands-on labs without guidance
- Ensure you are familiar with the Portal, PowerShell, and CLI
- Take and pass the practice exam multiple times
- Make use of the flashcard deck
- Participate in the Linux Academy community and Slack channel