Intro

This document compiles the transcripts of most of the demos contained in the following courses:

- https://app.pluralsight.com/ilx/microsoft-certified-azure-administrator-associate-(az-104)-deploy-and-manage-azure-compute-resources
- https://app.pluralsight.com/ilx/microsoft-certified-azure-administrator-associate-(az-104)-manage-azure-identities-and-governance
- https://app.pluralsight.com/ilx/microsoft-certified-azure-administrator-associate-(az-104)-monitor-and-maintain-azure-resources
- https://app.pluralsight.com/ilx/microsoft-certified-azure-administrator-associate-(az-104)-implement-and-manage-storage
- https://app.pluralsight.com/ilx/microsoft-certified-azure-administrator-associate-(az-104)-implement-and-manage-virtual-networking

These notes are NOT sufficient to replace the aforementioned courses, but they can be a handy aid during a training journey.

The interface and functionalities of Microsoft Azure are continuously evolving, therefore some information reported here might not be accurate.

At the following link, you can find the files in markdown. They can be modified at will and you can request a commit to keep the repository updated:

https://github.com/EliaFeltrin/az104_howTo/tree/main

The author disclaims any responsibility should the information contained herein be incomplete, inaccurate, or incorrect.

Table of contents

Definitions

Azure Resource Group: A logical container that holds related Azure resources, making it easier to manage and organize them based on their lifecycle or deployment.

Azure Virtual Machine (VM): A scalable computing resource that provides an on-demand, virtualized environment for running applications, development, and testing.

Azure Availability Zone: A physically separate zone within an Azure region, providing high availability by protecting resources from data center failures.

Azure Availability Set: A logical grouping that ensures virtual machines are distributed across different physical servers to reduce the risk of downtime.

Azure App Service Plan: The compute resources for an Azure App Service, defining the region, size, and scale of the web applications hosted.

Azure App Service: A fully managed platform for building, deploying, and scaling web apps, APIs, and backend services.

Azure Web App: A specific type of Azure App Service optimized for hosting web applications and APIs.

Azure Container: A lightweight, portable, and self-contained runtime environment for running applications and their dependencies.

Azure Container Instance: A managed service for running containers directly in the Azure cloud without requiring virtual machine management.

Azure Container App: A platform for deploying microservices and containerized applications in a serverless environment, supporting features like scaling and ingress.

Azure Virtual Network (VNet): A private network in Azure that allows resources to securely communicate with each other, the internet, and on-premises networks.

Azure Virtual Subnetwork (Subnet): A segment of an Azure VNet that isolates and organizes resources for improved security and routing.

Azure Resource Manager (ARM) Template: A JSON file that defines infrastructure and configurations for Azure resources in a declarative manner.

Azure Bicep: A domain-specific language for deploying Azure resources, offering a simpler and more readable syntax compared to ARM templates.

Azure Lock: A mechanism to protect resources from accidental deletion or modification by applying either a "ReadOnly" or "Delete" lock.

Azure Assign Policy: A rule-based mechanism to enforce organizational standards by applying policies to Azure resources.

Azure Management Group: A container for managing access, policies, and compliance across multiple Azure subscriptions.

Azure Storage Account: A scalable and durable service for storing data in Azure, supporting blobs, tables, queues, and files.

Azure Monitor Log: A feature of Azure Monitor that collects and stores log data for querying and analyzing resource activity.

Azure Data Collection Endpoint: A centralized endpoint for ingesting monitoring data into Azure Monitor.

Azure Data Collection Rule: A configuration that specifies how data is collected and routed to Azure Monitor logs or metrics.

Azure Alerts: Notifications or actions triggered when specified conditions are met in Azure Monitor.

Azure Actions: Tasks or operations executed in response to Azure Alerts, such as sending emails or invoking webhooks.

Azure Monitor Insight: Prebuilt dashboards and analytical tools for visualizing and analyzing Azure resource performance and health.

Azure Network Watcher: A tool for monitoring and diagnosing network connectivity issues in Azure VNets.

Azure Queue, Table, File, Blob: Components of Azure Storage:

- **Queue:** A messaging system for asynchronous communication between components.
- **Table:** A NoSQL datastore for structured, schemaless data.
- **File:** A fully managed file share accessible via SMB protocol.
- **Blob:** An object storage solution for unstructured data like documents and media.

Azure VNet Peering: A mechanism for connecting two Azure VNets, enabling private communication without using public internet.

Azure Network Security Group (NSG): A set of security rules that control inbound and outbound traffic to Azure resources in a VNet.

Azure Application Security Group (ASG): A logical grouping of virtual machines for managing security rules and isolating workloads.

Azure Load Balancer: A Layer 4 load balancer that distributes incoming network traffic across multiple Azure resources to ensure high availability.

Azure DNS: A service for hosting and managing domain name system (DNS) records in Azure.

Azure Bastion: A managed service that provides secure and seamless RDP and SSH connectivity to Azure virtual machines without exposing them to the public internet.

Azure Service Endpoint: A feature that extends a VNet's private address space to Azure services, enhancing security by restricting access to a VNet.

Azure Resource Manager (ARM): The deployment and management framework for Azure resources, enabling consistent management through templates, APIs, and CLI tools.

Azure Resource Provider: A service in Azure that provides resources, such as virtual machines, storage, or databases, for a specific Azure service.

Azure Tenant: A dedicated instance of Azure Active Directory (AAD) associated with an organization, used to manage identities and access.

Microsoft Entra Identities: Cloud-based identity management services that include Azure AD, enabling secure sign-ins and access control for users and devices.

Azure Groups: Collections of users or devices in Azure AD for managing access and assigning permissions.

Azure Licenses: The entitlements that determine which features and services users in Azure AD can access.

Azure Roles: Role-based access control (RBAC) definitions that determine permissions for managing Azure resources.

Deploy and Manage Azure Compute Resources

Deploy a basic VM

- **Azure portal**
  - Enter a resource group > Create > Search for virtual machine > Create
    - Set name, region, availability option, set the image
    - Set up the authentication method you prefer
    - Set up disk
    - Create a new data disk (NB: it's different from the OS disk) if needed
      - Set up name, size, and type
    - Create
- **Bash**
  - `az vm create --name [name] --resource-group [resource group name] --image [image] --security-type [Standard | TrustedLaunch | Confidential] [--authentication-type password --generate-ssh-keys | --authentication-type ssh --admin-username --admin-password]`
  - NB: `az vm --help` to get all flags

Resize a VM

- Enter the VM > Availability + Scale (sidebar) > Size
  - Choose the new size > Resize
- NB: If the VM is running, you cannot resize to all the possible sizes. Stop it if you want more options.
- NB: Resizing makes the resource temporarily unavailable.

Create, attach, and partition a disk

- Enter a resource group > Create > Search for managed disk > Create
  - Set name, region, zone, and size
  - Set other parameters if needed, otherwise use the default
  - Create
- Enter the virtual machine > Settings (sidebar) > Disks > Attach existing disk
  - Select the new one
- Connect to the virtual machine via SSH (assuming Linux):
  - Run `lsblk` to list the mounted drives
  - Run `sudo parted /dev/[name of the disk] --script mklabel gpt mkpart [start %] [end %]`
  - Run `sudo mkfs.xfs -f /dev/[new partition name]` to format it
  - Run `sudo partprobe /dev/[new partition name]`

Encrypt VM disk

- **PowerShell**
  - Create a key vault
    - `New-AzKeyVault -Name [vault name] -ResourceGroupName [rg name] -Location [vault location] -EnabledForDiskEncryption`
  - Save vault details in a variable
    - `$keyvault = Get-AzKeyVault -VaultName [vault name] -ResourceGroupName [rg name]`
  - Encrypt the disk
    - `Set-AzVmDiskEncryptionExtension -ResourceGroupName [rg name] -VMName [vm name] -DiskEncryptionKeyVaultUrl $keyvault.VaultUri -DiskEncryptionKeyVaultId $keyvault.ResourceId [-SkipVmBackup] -VolumeType All`

Create a VM across availability zones

- Enter the RG > Create > Search for virtual machine > Create
  - Set up all the other params
  - Set availability options to availability zones
  - Choose the AZs you want in the availability zones menu
  - Next (networking) > Create a load balancer (end of the page)
    - Set name and protocol
    - Create
  - Create

Create a VM in an availability set

- Enter the RG > Create > Search for virtual machine > Create
  - Set up all the other params
  - Set availability options to availability set
  - Create new under availability set menu
    - Set name, fault domains, and update domains
    - OK
  - Create
  - You can now create other VMs inside the same availability set

Configure VM scale set demo

- Enter the RG > Search for scale set > Create
  - Set name, region, orchestration mode, image, size, authentication method
  - Next (networking) > Choose a VNet > Set load balancing option to Azure load balancer > Create a load balancer
    - Set name, type, ...
    - Create
  - Next (scaling) > Set initial, minimum, maximum number of instances
    - Set threshold for scaling in/out and scale policy
  - Next (advanced)
    - Set allocation policy

            - Copy into the custom data section this script https://learn.microsoft.com/en-us/azure/virtual-machines/linux/tutorial-automate-vm-deployment#create-cloud-init-config-file
        - Create
- You can enter the load balancer public address via browser and test the scale set

Create an app service plan

- Enter a resource group > Create > Search app service plan > Create
    - Set name, operating system, ..., pricing plan
    - Create

Create an app service web app demo

- Enter a resource group > Search for web app > Create
    - Set a globally unique name
    - Set publish to code
    - Set runtime stack to Python 3.12
    - Set operating system to Linux
    - Set region
    - Set the app service plan
    - Next (networking) > Enable public access
    - Create
- Enter the web app
    - Copy the default domain
    - Try to access it through your browser
- Add a custom domain name
    - Enter the web app > Settings (sidebar) > Custom domains > Add a custom domain
        - Set up all configs
        - Set add a certificate later under TLS/SSL certificate
        - Create new under app service domain
            - Set the new domain
        - Next (advanced) > Disable auto-renewal
        - Create
    - Go back to the web app page > Settings (sidebar) > Custom domain > Add binding in the new domain row
        - Choose create app service managed certificate under source
        - Add
- Access restriction
    - Go back to the web app page > Settings (sidebar) > Networking > Click on the link in the public network access row
    - Choose your restriction rules
    - NB: Here you can also set up private endpoints, virtual network integrations, and hybrid connections
    - In the settings panel, you can also set up scale up and out rules, backups, and restore from backups
- Enter Deployment (sidebar) > Deployment slots > Add slot

    - Set name
    - Set clone setting method
- Enter Deployment (sidebar) > Deployment center
    - Choose external Git as source
    - Set repository to https://github.com/pluralsight-cloud/hello-python-app.git
    - Choose the main branch
- You can try to access the web app, restore a previous backup, swap deployment slot

Build and publish containers

- Enter a resource group > Search for container registry > Create
    - Choose a globally unique name
    - Create
- Enter the container registry > Settings (sidebar) > Enable admin user
- Open PowerShell
    - Run `git clone https://github.com/pluralsight-cloud/Microsoft-Certified-Azure-Administrator-Associate.git`
        - CD into the introduction to containers folder
        - `az acr build --image sample/hostnameapp:v1 --registry [previously created container registry name] --file ./Dockerfile .`
        - `az acr run --registry [previously created container registry name] --cmd '$Registry/sample/hostnameapp:v1' /dev/null`

Create Azure container instances

- Enter the resource group > Search for container instances > Create
    - Set name, region, ...
    - Set image source to Azure container registry
    - Choose your previously created container registry, image, and tag
    - Choose the size
    - Next (networking) > Choose the container type (public/private), ...
    - Next (advanced) choose the restart policy
    - Create
- Enter the container instances page
    - Copy the public IP address and access it

Create a container app

- Enter a resource group > Search for container app > Create
    - Set name, region, container environment
    - Next (container) > Choose your previously created registry, image, and tag
    - Create
- Enter the container app > Settings (sidebar) > Ingress
    - Enable ingress
    - Choose accepting traffic from anywhere
    - Allow insecure connection
    - Set target port to 80
    - Save

- Go back to the overview page > Try to access the application through the link
- Click on Application (sidebar) > Scale > Edit and deploy
    - You can add another application (container image > add)
    - Next (scale) > Set up scaling limit (down to 0 -> no costs)
        - Set up scaling rules

Manage Azure Identities and Governance

Create Virtual Network

- **Bash**
    - `az network vnet create --name [vnet name] --resource-group [resource group name] --address-prefix [address prefix]`
        - Example: `az network vnet create --name myvnet --resource-group $rgname --address-prefix 10.0.0.0/16`
- **PowerShell**
    - `New-AzVirtualNetwork -Name [vnet name] -ResourceGroupName [rg name] -Location [rg location] -AddressPrefix "[address prefix]"`
        - Example: `New-AzVirtualNetwork -Name myvnet -ResourceGroupName $rgname -Location $rglocation -AddressPrefix "10.0.0.0/16"`

Create a Resource via ARM Template

- Search for "deploy" > "Deploy a custom template" > "Build your own template in the editor"
    - Add all resources you need
    - Optionally export the template
    - Deploy

Export a Template from an Existing Resource Group

- Enter the resource group > Automation (sidebar) > Export template

ARM Template to Bicep

- **Bash**
    - `az bicep decompile --file [input file.json]`

Compile Bicep

- **Bash**
    - `az bicep build --file [input file.bicep] --outfile [output file.json]`

Move Resource

- Select the resources you want to move inside a RG > "Move" (top panel) > Choose the option you prefer
    - Set the target

Lock Resource

- Enter a resource > Settings (sidebar) > Locks > Add
    - Set name
    - Set type

Assign Policy

- Search for policy > Authoring (sidebar) > Definitions
    - Search for the policy you want to use
    - Assign policy
        - Set parameters
        - Set the non-compliant message

Use Management Groups

- Search for management group > Start using management group
    - Set ID and name
- You can create management groups up to 6 levels of depth
- Select a management group > Access control (sidebar)
    - Add > Add role assignments
        - Configure with people and roles
- Select Policy (sidebar) > Assign policy
- Select Budget (sidebar)

Managing Dynamic Groups

- Select Groups (sidebar)
- Add new group
    - Set name and description
    - Set the membership to dynamic user
    - Create your query
    - Double-check with the preview

Monitor and Maintain Azure Resources

Configure Azure Monitor Logs for a Storage Account

- Enter the desired resource group
- Create a storage account from the marketplace
    - Set a sandbox-unique name
    - Annotate the region, needed to configure the Log Analytics workspace
- Create a Log Analytics workspace
    - Set the same region as the storage account
- Go to the resource > Monitoring (sidebar) > Diagnostic settings
- Click on the storage account > Add diagnostic setting
    - Set a name

        - Select the metrics you want to collect
        - Select send to Log Analytics workspace and select the right one
- Click on Save
- Click on Blob > Add diagnostic setting
    - Set a name
    - Select the logs/metrics you want to collect
    - Select send to the Log Analytics workspace and select the right one
- Test:
    - Inside the storage account, select Data Storage (sidebar) > Containers
        - Click on the + in the upper left and set the name for the new container
        - Enter the container and upload some files
    - Search Monitor in the search bar
        - Click on Logs
        - Select the storage account
        - Use a pre-defined query or write your own

Set Up Data Collection Endpoint, Data Collection Rule for a VM

- Create a virtual machine inside the resource group
- Enter the resource group and create a Log Analytics workspace
    - Choose a name
    - Select the same region as the virtual machine
- Search Monitor, go to Settings (sidebar) > Data Collection Endpoint > Create
    - Set the same region as the VM
- Search Monitor, go to Settings (sidebar) > Data Collection Rule > Create
    - Set name
    - Select the same resource group as the VM
    - Select the same region as the VM
    - Select the OS of the VM
    - Next (Resources) > Add resources > Select your VM
    - Next (Collect and Deliver) > Add data source > Select what you want to collect > Select the destination
    - Create
- Inside Monitor > Logs (sidebar)

Setting Up Alerts and Actions

- Search Automation Account > Create
    - Select the desired resource group
    - Select the same region as the VM you want to control
    - Create
- Enter the VM you want to control > Monitoring (sidebar) > Alerts > Create > Alert rule
    - Select the desired triggering signal > Apply > Next (Action)
        - Select Use Action Group > Create Action Group (right panel)
            - Select the right resource group
            - Leave region = Global
            - Set the name and display name

        - Next (Notification) > Configure notifications (optional)
        - Next (Action) > Select Automation Runbook > Complete Runbook configuration (right panel)
        - Create
    - Next (Details) > Enter alert rule name
    - Create

Azure Monitor Insights

- Create a resource group, a VM, a storage account, and a Log Analytics workspace
- Enter the virtual machine > Monitoring (sidebar) > Insights > Enable
    - Select the data collection rule or create a new one
- Click on Azure Monitor in the top bar (inside VM's Insights) > Configure Insights > Monitored
    - Select the VM > Performance/Map

Network Watcher

- Prerequisites: Two resource groups in the same region with a VM, a VNet, a storage account, and a Log Analytics workspace, a peer-to-peer connection between the two VMs
- Search for Network Watcher > Create
    - Select the region you want
    - Apply
- Click on Monitoring (sidebar) > Connection Monitor > Create
    - Set name and the same region as virtual machines
    - Next (Test Group) > Add sources > One of the resource groups > VNet > VSubnet > VM > Add endpoints > Apply
    - Add test configuration > Select ICMP protocol > Set thresholds > Add test configuration
    - Add destination > Select the other resource group > VNet, VSubnet > VNet
    - Next (Workspace) > Select the workspace
    - Create
    - Go to Connection Monitor to see results
- Click on Network Diagnostic Tool (sidebar) > NSG Flow Logs
    - Select protocol (TCP)
    - Copy and paste the source VM private IP address (something like 10.2.0.4)
    - Copy and paste the destination VM private IP address
    - Set destination port (445 for TCP)
- Click on Network Diagnostic Tool (sidebar) > Packet Capture > Add
    - Select the virtual VM inside with the storage account
    - Start
- Refresh
- Wait some time > Click 3 dots > Stop capture
- You can now download the .cap file and open locally with Wireshark to check what's happening
- Click on Network Diagnostic Tool (sidebar) > Effective Security Rules
    - Select the VM
    - You can investigate the security rules affecting the VM

Implement and Manage Storage

Access an Azure Queue/Table/File/Blob Resource

- `https://[storage account].[type of resource].core.windows.net`

Storage Redundancy Options:

- **Local Redundant Storage (LRS):** 3 copies in the same availability zone
- **Zone Redundant Storage (ZRS):** one copy for each zone in the region
- **Geo Redundant Storage (GRS):** 3 in the same zone in two different regions
- **Geo Zone Redundant Storage (GZRS):** 3 copies spread in 3 zones in the home region, and 3 copies in one zone in another region
- **Read Access Geo Redundant Storage (RA-GZRS):** same as GZRS but the copies in the foreign region are read-only

Create a Storage Account

- Search for "storage" > Storage account
- Select/create a resource group
- Set name, region, performance, redundancy
- Create

Choose Your Blob Storage Tier

```
Access cost         ^
    |               |   Hot: Frequently accessed
    |               |   Cool: Accessed once in 30 days
    |               |   Cold: Accessed once in 90 days
    |               |   Archive: Accessed once in 180 days
    |               |
    v          Storage cost
```

Set Up a Blob

- Enter the storage account > Data Storage (sidebar) > Containers (AKA blobs)
- Click on Create > Set name > Apply > Enter the new container > Upload
    - In advanced options, you can set additional parameters such as type, block size, access tier
- Go back to the storage account > Data Management (sidebar) > Data Protection > Tracking > Enable versioning for blobs
    - Set the behavior
    - Save

Configure Object Replication

- Create two storage accounts (even in different regions)
- Enter the source blob > Data Management (sidebar) > Data Protection > Tracking
    - Enable versioning for blob

    - Enable blob change feed
    - Save
- Enter the destination blob > Data Management (sidebar) > Data Protection > Tracking
    - Enable versioning for blob
    - Save
- In the destination blob > Data Storage (sidebar) > Containers > Create the destination container
- In the source blob > Data Storage (sidebar) > Containers > Create the source container
- Enter the source container and upload some files
- In the source container > Data Management (sidebar) > Object Replication > Create object replication rule
    - Set destination storage account, source and destination containers
    - Optionally add filters (NB: use / at the end of a folder name if you want files inside that folder)
    - Set the copy over
    - Create

Configuring Lifecycle Management

- Inside the storage account > Data Management (sidebar) > Lifecycle Management > Add a rule
    - Set name, scope, type, subtype
    - Next (Base Blobs) > Set the rules
    - Next (Filter Set) > Set the prefix (NB: remember / at the end of a path)

Configure Azure Files

- Enter the storage account > Data Storage (sidebar) > File Share > Create
    - Set name, tier
    - Set backup (eventually)
    - Create
- Upload some files
- Connect (top panel) > Set target OS
    - Set authentication method
    - Copy the script
    - Execute the script on the target machine

Managing Azure Files (Snapshots)

- Enter the storage account > Data Storage (sidebar) > File Shares
    - Check that soft delete option is ENABLED (top panel)
    - Eventually remove the lock placed by Azure Backup running in the background to provide soft deletion
- Enter the file share > Operations > Snapshots > Add Snapshot

Providing Access to Azure Files

- Enter the storage account > Data Storage (sidebar) > File Shares
- Click on the "Identity-based access" label (top panel)
    - Choose the identity source and set the desired parameters
    - Save

- Access Control (sidebar) > Setup roles

Implement and Manage Virtual Networking

Create a Virtual Network

- Search for Virtual Network in the marketplace > Create
  - Set name, region, and so on
  - Next (IP addresses) > Set the address space > Add subnet
  - Create

Make a Virtual Machine Reachable from Public Net Through SSH

- Create a public IP for VM1
  - Search for Public IP > Create
    - Set name, version, ...
    - Set routing preference to Microsoft network
    - Create
  - Enter the public IP address > Associate (top panel)
    - Set network interfaces to VM1's network interface
- Enter VM1 > Networking (sidebar) > Network settings > Add network security group
  - Create port rule > Inbound port rule
    - Set service to SSH
- NB: Now you can enter VM1 via SSH, type `hostname -I` to get its private IP, and connect to VM2 adding 1 to that private IP address -> VM1 results in a jump-in machine

Routing Traffic into a VNet

- Search for Route Table > Create
  - Set region, name, and resource group
  - Set "Propagate gateway routes" to "No" prevents propagation of on-premises routes to the network interfaces in associated subnets
  - Create
- Enter the resource > Settings > Routes (sidebar) > Add
  - Set destination type to IP addresses to 0.0.0.0/0 (meaning it captures all traffic)
  - Set next hop type to virtual appliance and its IP
- Click on Subnets (sidebar) > Associate
  - Choose the virtual network and the subnet
- Entering your virtual machine in the subnet > Networking > Network setting > Network interfaces > Help (sidebar) > Effective routes
  - You can check that the system default routes that connect the VM to the internet is overridden by our rule

Configure VNet Peering (Remember it is not transitive)

- Enter VNet1 > Settings > Peering > Add
  - Set both side peering name

    - Select VNet2 as virtual network
    - Set the options you need from both sides

Securing VNets with Network Security Groups (NSGs) for Nginx

- Search for Network Security Group on the marketplace
    - Set name and region
    - Create
- Enter the Network Security Group > Settings (sidebar) > Network interfaces
    - Open a Cloud Shell > SSH `[username]@[NIC public IP]`
    - `sudo apt update && sudo apt install nginx -y`
- Go back to the NSG > Settings (sidebar) > Inbound security rules > Add
    - Set source and destination to Any
    - Set service to HTTP
    - Add
- Check from browser: the Nginx page should be reachable

Extending NSGs with Application Security Groups (ASGs)

- Search for Application Security Group in the marketplace > Create
    - Choose resource group, set name, and region
    - Create
- Enter one by one the VMs you want to associate with the security group > Networking (sidebar) > Application Security Group > Add Application Security Group
    - Select the ASG
    - Add
- Make sure that the VMs are not associated with some NSG individually. Eventually, delete this association
- Enter the security group > Settings (sidebar) > Subnets > Associate
    - Choose the virtual network your VMs are in and the subnet
- Now you can create/update rules in the NSG setting as destination an ASG

Implement Azure Load Balancer

- Search for Load Balancer in the marketplace > Create
    - Set name
    - Set region to the same as the VNet you are working in
    - Set SKU according to your scenario
    - Set Type to Public
    - Set tier (Regional if all your VMs are in the same region, Global otherwise)
    - Next (Frontend IP config) > Add a frontend IP config
        - Name the IP config, set version and type
        - Add a public IP address
            - Name it
            - Save
        - Save
    - Next (Backend pools) > Add backend pool
        - Name it

        - Set the virtual network you are working in
        - Add your resource through IP address or NIC
    - Next (Inbound rules)
        - Add a load balancing rule
            - Name it
            - Set IP version
            - Set your previously created IP config as frontend IP address
            - Set your previously created backend pool
            - Choose the protocol, the frontend (public) and backend (private) port
            - Health probe: create new
                - Name it
                - Set it coherently with what the pool instance is expected to do
                - Save
            - Save
        - Add an inbound NAT rule (to forward SSH connections)
            - Name it
            - Set type to Backend pool
            - Set the backend pool to the previously created ones
            - Set the frontend IP address through the previously created IP config
            - Choose a non-conflicting port as frontend port range start (e.g., 1000)
            - Set the backend port as the one you will use for SSH in your backend
            - Save
    - Next (Outbound rules)
        - Set it if you plan to reach an external server from your VMs
    - Create

Implement Private DNS

- Search for Private DNS Zone in the marketplace > Create
    - Set the name you want high-level domain (i.e., contoso.com, then you will be able to set up vm1.contoso.com, vm2.contoso.com, and so on)
    - Create
- Enter the Private DNS Zone > Settings (sidebar) > Virtual Network Links > Add
    - Set the name and the virtual network you want to connect to
    - Tick enable auto-registration to connect each resource inside the VNet to the DNS

Connect Using Azure Bastion

- Enter the VNet where the subnet you want to jump in through Azure Bastion is > Add a subnet
    - Choose Azure Bastion as subnet purpose
    - Assign a /26 or larger address space
    - Save
- Go back to the resource group > Create > Search for Azure Bastion > Create > Set up manually (or deploy using the default config if they match your needs)
    - Name it
    - Set the same region of the VNet you want to access
    - Set the VNet and the Bastion subnet you have previously created

    - Set up the Bastion public IP
    - Eventually set up advanced options
    - Create
- Test: Enter a VM in a subnet inside the Bastion VNet > Connect (top panel) > Connect with Bastion > Enter your credentials/key for the VM

Privately Integrating Public Services (Service Endpoints)

- Create a storage account as an example
- Enter the VNet where the VM that needs to connect to a service is
- Select the subnet > Edit > Tick Microsoft.Storage in the Service menu under Service endpoints > Save
- Enter the storage account > Security + Networking (sidebar) > Networking > Private Endpoint Connections (top panel) > Add Private Endpoint
    - Set name, ...
    - Next (Resource) > Choose the target subservice
    - Next (Virtual Network) > Set VNet and subnet and choose IP config (if you will use DNS dynamically it's ok)
    - Next (DNS Zone) > Setup
    - Create

Restrict Public Access to a Storage Account

- Enter the storage account you want to secure > Security + Networking (sidebar) > Networking > Firewalls and Virtual Network (top panel) > Select Enabled from selected VNet and IP addresses
    - Set up