Caltech | Center for Technology & Management Education

Post Graduate Program in Cloud Computing

Powerd by simplilearn

Cloud Computing

Caltech | Center for Technology & Management Education

PG CC - Microsoft Azure Architect Design: AZ:304

Powerd by simplilearn

Design a Compute Solution

Learning Objectives

By the end of this lesson, you will be able to:

- Recommend a solution for Compute Provisioning
- Determine Appropriate Compute Technologies
- Recommend a solution for Containers
- Recommend a solution for Automating Compute Management

A Day in the Life of an Azure Architect

You are working as an architect in an organization and you have been asked to design a solution to deploy an application for testing and production. The application should be deployed to different environments and should run without installation dependencies.

Your company is also looking for a solution that can provide the quickest and most straightforward approach to host a container in Azure.

To achieve all of the above, along with some additional features, we would be learning a few concepts in this lesson that will help you find a solution for the above scenario.

Recommend a Solution for Compute Provisioning

Choose an Azure Compute Service

Compute refers to the hosting model for the computing resources that application runs on.

Application may need multiple workloads, evaluate each workload separately.

As a result, it is possible that a complete solution may incorporate two or more compute services.

Choose an Azure Compute Service

There are two types of compute strategies:

Lift and shift

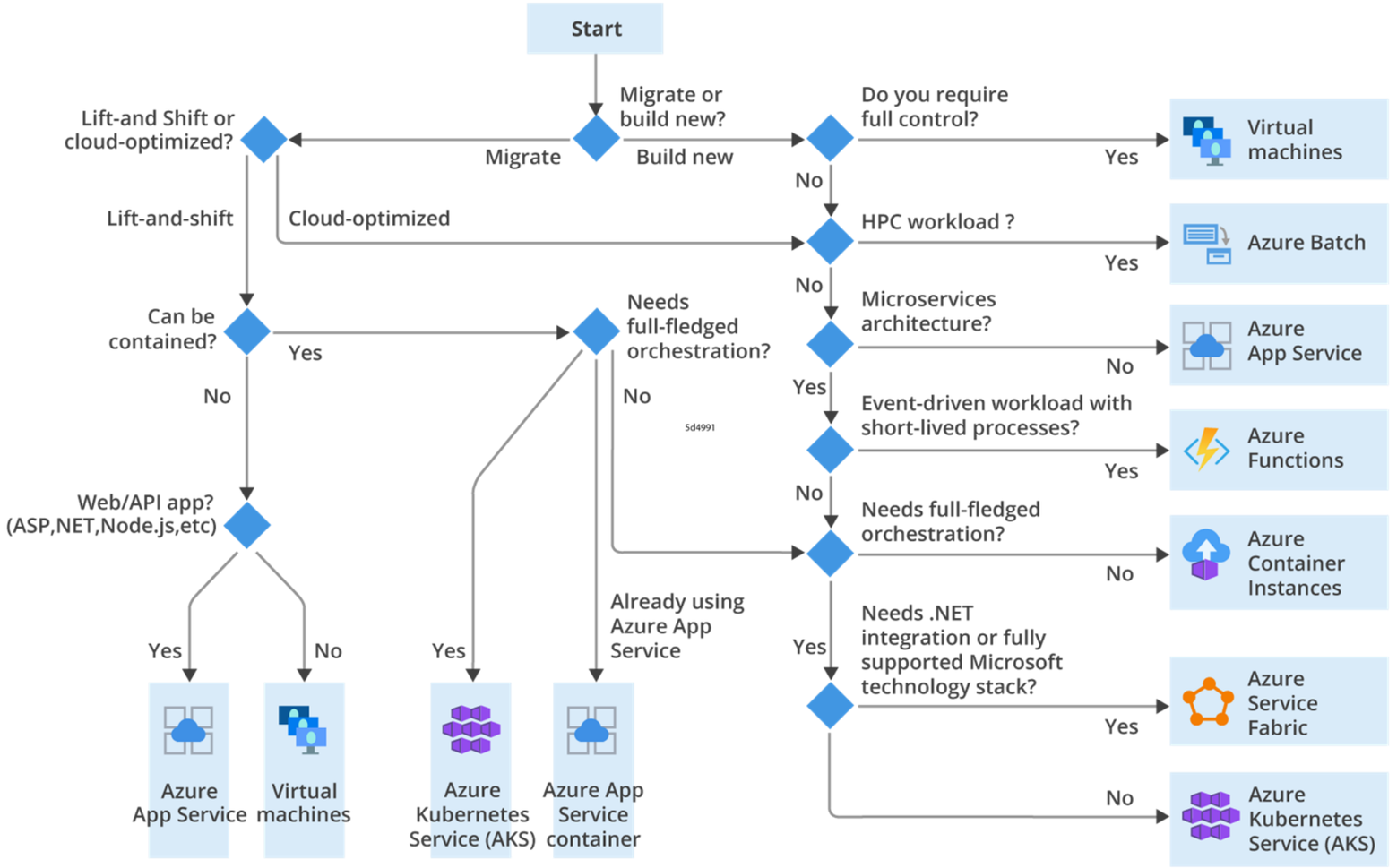
Migrating a service without redesigning the application or making code changes

Cloud optimized

Take advantage of cloud-native features and capabilities

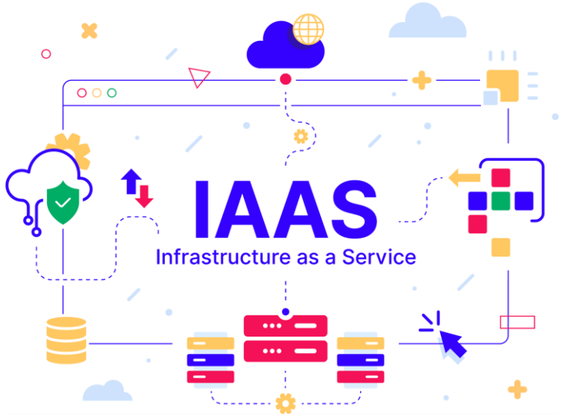
Choose a Candidate Service

This diagram shows how to choose a candidate service:



Hosting Models

There are two most common hosting models:



- It lets user provision individual VMs along with the associated networking and storage components.
- It gives the most control, flexibility, and portability.

- It is a complete development and deployment environment in the cloud, with resources.
- It enables the delivery of everything from simple cloud-based apps to sophisticated, cloud-enabled enterprise applications.

Candidate Services

These are the different candidate services:

App Service	A managed service for hosting web apps, mobile app back ends, RESTful APIs, or automated business processes
Functions	A managed PaaS service
Batch	A managed Kubernetes service for running containerized applications.
Azure Kubernetes Service	A managed service for running large-scale parallel and high-performance computing (HPC) applications

Candidate Services

These are the different candidate services:

Container Instances	The fastest and simplest way to run a container in Azure, without having to provision any virtual machines and without having to adopt a higher-level service
Virtual machines	Deploy and manage VMs inside an Azure virtual network
Azure Service Fabric	Package, deploy, and manage scalable and reliable microservices and containers

Service Limits and Cost

Every service has service limits applicable.

Criteria	Virtual Machines	App Service	Service Fabric	Azure Functions	Azure Kubernetes Service	Container Instances	Azure Batch
Application composition	Agnostic	Applications, containers	Services, guest executables, containers	Functions	Containers	Containers	Scheduled jobs
Density	Agnostic	Multiple apps per instance via app service plans	Multiple services per VM	Serverless	Multiple containers per node	No dedicated instances	Multiple apps per VM
Minimum number of nodes	1	1	5	Serverless	3	No dedicated nodes	1
State management	Stateless or Stateful	Stateless	Stateless or stateful	Stateless	Stateless or Stateful	Stateless	Stateless
Web hosting	Agnostic	Built in	Agnostic	Not applicable	Agnostic	Agnostic	No
Can be deployed to dedicated VNet?	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Hybrid connectivity	Supported	Supported	Supported	Supported	Supported	Not supported	Supported

It is important to consider Cost, SLA and Regional availability of the service.

Source: https://docs.microsoft.com/

Scalability of Compute Services

Guidance on service to be used for Scaling different resources:

Criteria	Virtual Machines	App Service	Service Fabric	Azure Functions	Azure Kubernetes Service	Container Instances	Azure Batch
Autoscaling	Virtual machine scale sets	Built-in service	Virtual machine scale sets	Built-in service	Pod auto-scaling[1], cluster auto-scaling[2]	Not supported	N/A
Load balancer	Azure Load Balancer	Integrated	Azure Load Balancer	Integrated	Azure Load Balancer or Application Gateway	No built-in support	Azure Load Balancer
Scale limit	Platform image: 1000 nodes per scale set, Custom image: 600 nodes per scale set	30 instances, 100 with App Service Environment	100 nodes per scale set	200 instances per Function app	100 nodes per cluster (default limit)	20 container groups per subscription (default limit).	20 core limit (default limit).

Availability of Compute Services

Guidance on service to be used for Availability of different resources:

Criteria	Virtual Machines	App Service	Service Fabric	Azure Functions	Azure Kubernetes Service	Container Instances	Azure Batch
Multi region failover	Traffic manager	Traffic manager	Traffic manager, Multi-Region Cluster	Azure Front Door	Traffic manager	Not supported	Not Supported

Assisted Practice

Creating a Virtual Machine

Duration: 10 Min.

Problem Statement:

As an Azure Architect, you've been tasked with recommending an Azure computing solution that provides an isolated environment for operating its OS and apps without relying on the underlying host system. YOur company will use this solution to deploy an application for testing and production.

Assisted Practice: Guidelines

Steps to create a virtual machine are:

1. Login to your Azure portal
2. Create a virtual machine on the Azure portal
3. Select Virtual machines
4. Create a new resource group

Unassisted Practice

Manage VM Sizes

Duration: 10 Min.

Problem Statement:

As an Azure Architect, you've been tasked with demonstrating the various VM sizes available in Azure for running your apps and workloads.

Unassisted Practice: Guidelines

Steps to manage VM size are:

1. Login to your Azure portal
2. Search and select virtual machines
3. Manage VM size

Unassisted Practice

Virtual Machine Extension: Powershell DSC **Duration: 10 Min.**

Problem Statement:

As an Azure Architect, you've been requested to recommend a solution for automating and configuring Azure VMs once they've been deployed. For example, we'll need that tool if a virtual machine requires software installation, anti-virus protection, or the execution of a script.

Unassisted Practice: Guidelines

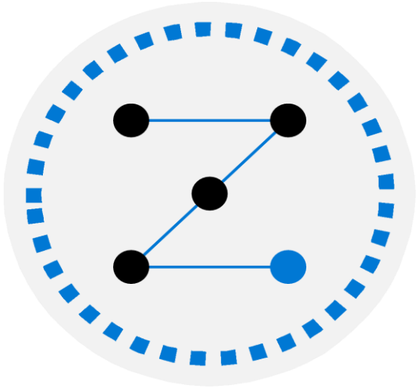
Steps to create virtual machine extension: Powershell DSC are:

1. Login to your Azure portal
2. Search and select virtual machines
3. Click on Extensions
4. Configure machine extension for a VM

Determine Appropriate Compute Technologies

Choosing an Azure Compute Option for Microservices

Below are the popular approaches for a Microservices architecture:



A service orchestrator that manages services running on dedicated nodes (VMs).

A serverless architecture using functions as a service (PaaS).

Service Orchestrators

An orchestrator handles tasks related to deploying and managing a set of services.

Service orchestration

Tasks include :

- Placing services on nodes
- Monitoring the health of services
- Restarting unhealthy services
- Load balancing network traffic across service instances
- Service discovery
- Scaling the number of instances of a service
- Applying configuration updates

Containers with Orchestration

Consider below orchestration services on the Azure Platform:

Azure Kubernetes Service (AKS):

AKS provisions Kubernetes and exposes the Kubernetes

API endpoints

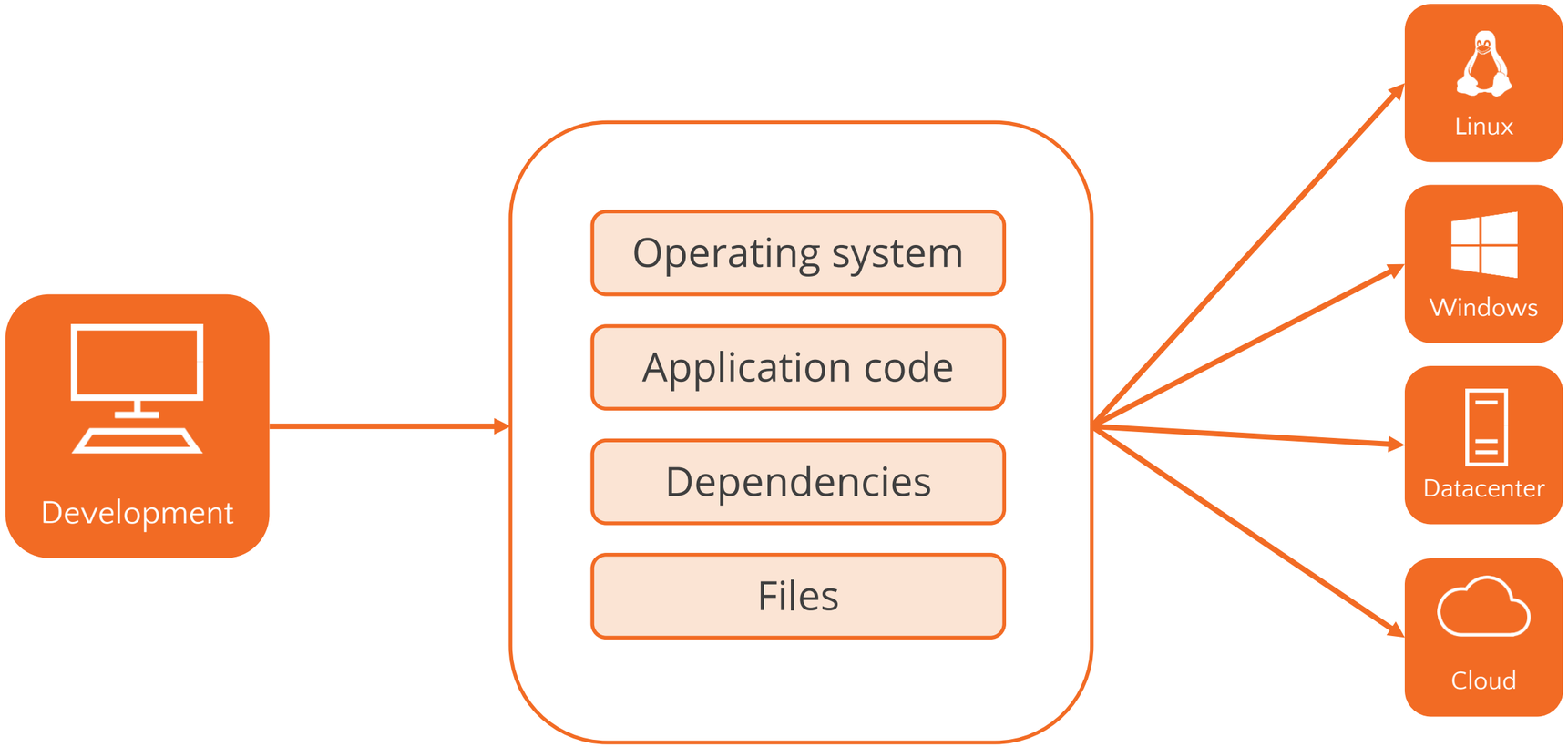
Service Fabric

Distributed systems platform for packaging, deploying, and managing microservices

Note: Docker Enterprise Edition and Mesosphere DC/OS can run in an IaaS environment on Azure.

Containers

A container is a loosely isolated environment that allows us to build and run software packages.



Containers

Container offers benefits which are specifically relevant to microservices:

Portability:	A container image is a standalone package that runs without needing to install libraries or other dependencies.
Agility:	Containers can be started and stopped quickly spin up new instances to handle more load or to recover from node failures.
Density:	Packing multiple containers onto a single node is useful when the application consists of many small services.
Resource isolation:	Limit the amount of memory and CPU that is available to a container to ensure that a runaway process doesn't exhaust the host resources.

Serverless

With a serverless architecture:

- User doesn't manage the VMs or the virtual network infrastructure.
- User deploys code and the hosting service handles putting and executing that code onto a VM.

This approach is suits small granular functions that are coordinated using event-based triggers.

Serverless

Azure Functions is a serverless compute service that supports various function triggers, including:

- HTTP requests
- Service Bus queues
- Event Hubs events

Example: A message being placed onto a queue might trigger a function that reads from the queue and processes the message.

Orchestrator vs Serverless

These are the difference between orchestrator and serverless:

- Manageability
- Flexibility and control
- Portability
- Application integration
- Cost
- Scalability

- Serverless apps are easy to manage as the platform manages all the compute resources.
- While, with an orchestrator, a user must keep an eye on load balancing, CPU, memory usage, and networking.

Orchestrator vs Serverless

These are the difference between orchestrator and serverless:

- Manageability
- Flexibility and control
- Portability
- Application integration
- Cost
- Scalability

- With serverless architecture users give up some degree of control because management details are abstracted.
- While orchestrator gives the user good control over configuring and managing user services and cluster.

Orchestrator vs Serverless

These are the difference between orchestrator and serverless:

- Manageability
- Flexibility and control
- Portability
- Application integration
- Cost
- Scalability

- Orchestrators can run on-premises or in multiple public clouds.

Orchestrator vs Serverless

These are the difference between orchestrator and serverless:

- Manageability
- Flexibility and control
- Portability
- Application integration
- Cost
- Scalability

- Building a complex application using serverless is difficult, one option is to use Azure Logic Apps to coordinate a set of Azure Functions.

Orchestrator vs Serverless

These are the difference between orchestrator and serverless:

- Manageability
- Flexibility and control
- Portability
- Application integration
- Cost
- Scalability

- With a Serverless app, the user pays for the actual compute resources consumed.
- While, with an orchestrator, the user pays for the VMs running in the cluster.

Orchestrator vs Serverless

These are the difference between orchestrator and serverless:

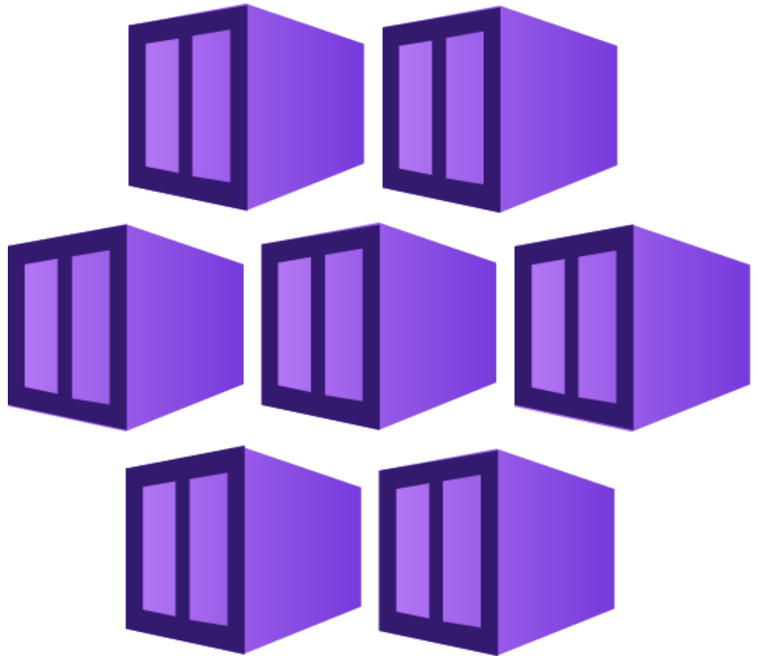
- Manageability
- Flexibility and control
- Portability
- Application integration
- Cost
- Scalability

- An orchestrator scales out by increasing the number of service instances running in the cluster, or, adding additional VMs to the cluster.
- While serverless like Azure functions scales automatically.

A Solution for Containers

Azure Kubernetes Service

Azure Kubernetes Service (AKS) manages your hosted Kubernetes environment and makes it simple to deploy and manage containerized applications in Azure.



Azure Kubernetes Service

The Azure Kubernetes Service provides following features:

Manages container-based applications

- It manages networking and storage requirements alongside
- It Focuses on application workloads instead of infrastructure components

Applications are described declaratively

- YAML files can be used to describe application
- Kubernetes handles management and deployment

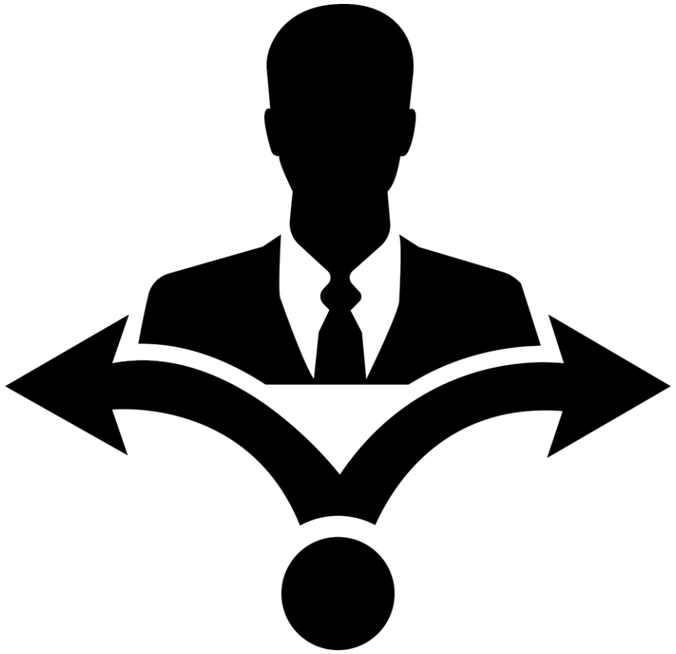
Makes it easier to orchestrate large solutions using a variety of containers

- Application containers
- Storage containers
- Middleware containers

When to use Azure Kubernetes Service

Users would approach the decision based on:

- Greenfield deployment, which allows users to evaluate AKS based on default features



- Lift and Shift Project, which forces users to consider features which best support their migration

When to use Azure Kubernetes Service

These are various features considered while deciding when to use Azure Kubernetes service;

Feature	Considerations and Decisions
Identity and Security	AD Integration
Logging and Monitoring	Azure Monitor for containers, custom monitoring solutions
Auto Scaling	Manual and auto, scheduled and programmatically, horizontal and vertical
Cluster Node Upgrade	Azure managed cluster upgrade and software updates
GPU Support	GPU-enabled node pools
Storage Volumes	Static and dynamic storage volumes
Virtual Network (VNet)	Deployed into existing VNet
HTTP Routing	Add-on HTTP application routing support
Docker Image	Default support for docker file image support
Private Container Registry	Integrates with ACR (Azure Container Registry), public and private repositories

Azure Container Instance

Simplest way to run a container in Azure:

- It doesn't require IaaS provisioning
- It doesn't require the adoption of a higher-level service

Ideal for one-off, isolated container instances:

- Simple applications
- Task automation
- Build jobs

Azure Container Instance

Azure Container Instance has the following features:

- It supports Linux and Windows containers
- It supports direct mounting of Azure Files Shares

A Container can be provisioned with a public IP address and DNS name.

When to use Azure Container Instances

Feature	Description
Fast startup times	Containers can start in seconds without the need to provision and manage VMs
Public IP connectivity and DNS name	Containers can be directly exposed to the internet with an IP address and a fully qualified domain name (FQDN)
Hypervisor-level security	Container applications are as isolated in a container as they would be in a VM
Custom sizes	Container nodes can be scaled dynamically to match actual resource demands for an application
Persistent storage	Containers support direct mounting of Azure Files shares
Linux and Windows containers	The same API is used to schedule both Linux and Windows containers
Co-scheduled groups	Container Instances supports scheduling of multi-container groups that share host machine resources
Virtual network deployment	Container Instances can be deployed into an Azure virtual network

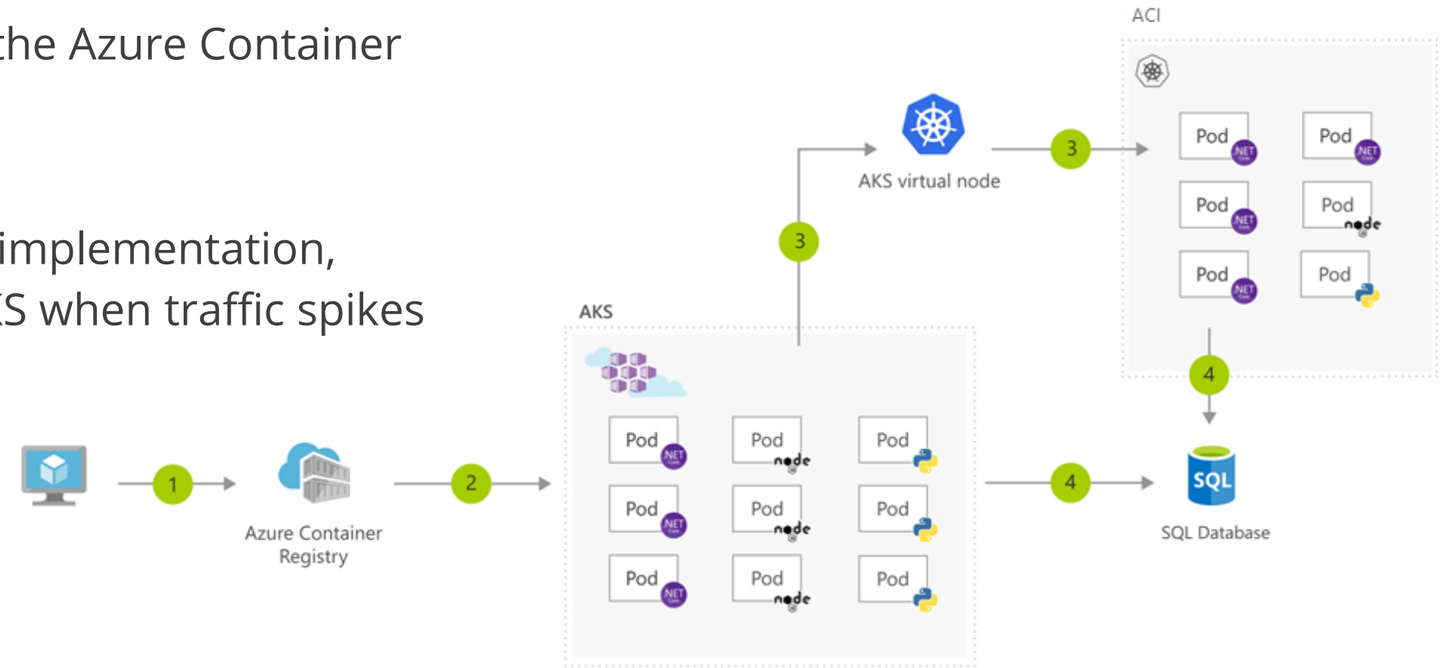
Recommendation

For scenarios where users need full container orchestration, including service discovery across multiple containers, automatic scaling, and coordinated application upgrades, use Azure Kubernetes Service (AKS).

Bursting from AKS with ACL

Data flow:

- User registers container in Azure Container Registry
- Container images are pulled from the Azure Container Registry
- AKS virtual node, a Virtual Kubelet implementation, provisions pods inside ACI from AKS when traffic spikes
- AKS and ACI containers write to shared data store



Source: https://docs.microsoft.com/

Assisted Practice

Create a Container Instance

Duration: 10 Min.

Problem Statement:

As an Azure Architect, you've been asked to provide your organization with an Azure solution that can provide the quickest and most straightforward approach to host a container in Azure, without the need to manage any virtual machines or adopt a higher-level service.

Assisted Practice: Guidelines

Steps to create a container instance are:

1. Login to your Azure portal
2. Click on Create a resource
3. Select Containers and click on Container Instances
4. Create a Container Instance

Assisted Practice

Create a Container Registry

Duration: 10 Min.

Problem Statement:

Demonstrate ACR, a private registry for Docker container images hosted on Azure.

Assisted Practice: Guidelines

Steps to create a container registry are:

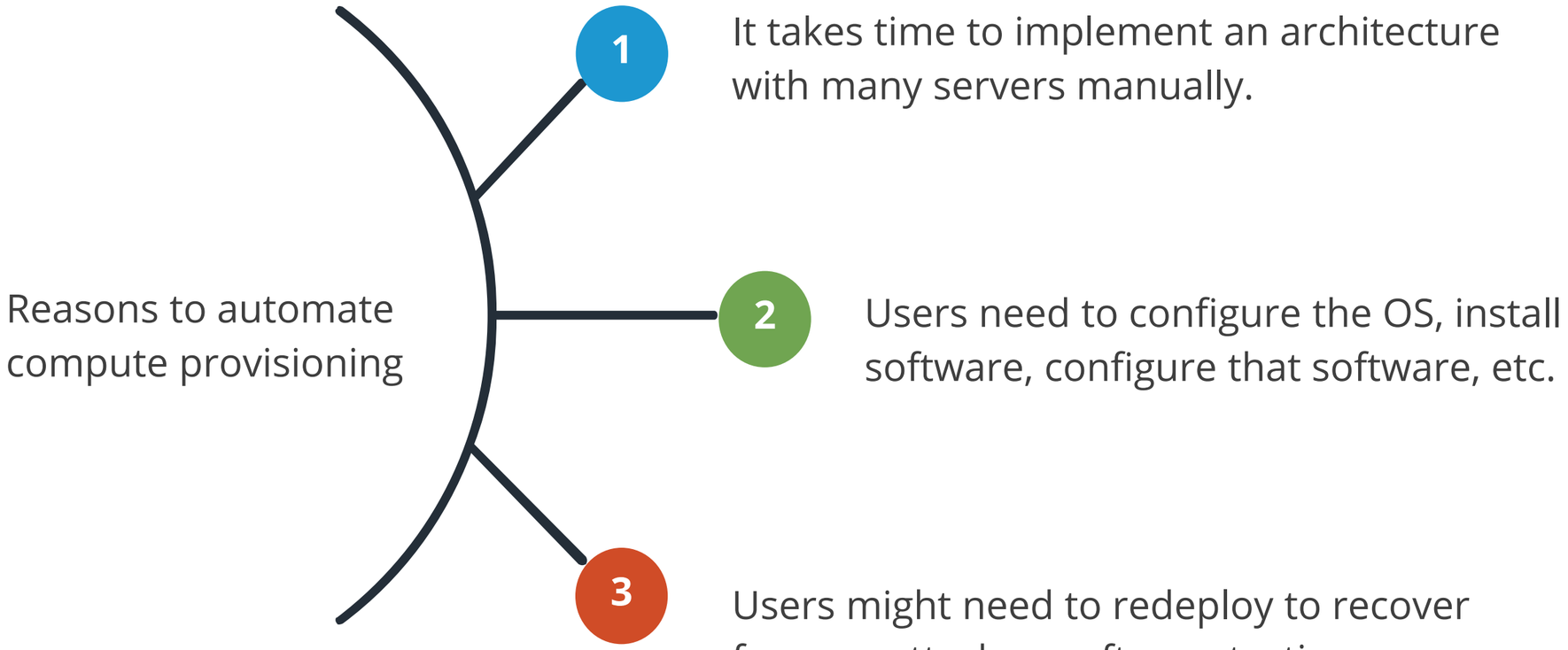
1. Login to your Azure portal
2. Click on Create a resource
3. Click on Containers, and select Container Registry
4. Create a container registry by providing the required details

Recommend a Solution for Automating Compute Management

Provisioning Solution for Azure Compute Infrastructure

Creating and managing compute resources manually requires a lot of time and is a repetitive task. As a result, users may want to automate the provisioning and management of compute resources.

Why Automate Compute Provisioning?



Automated Provisioning

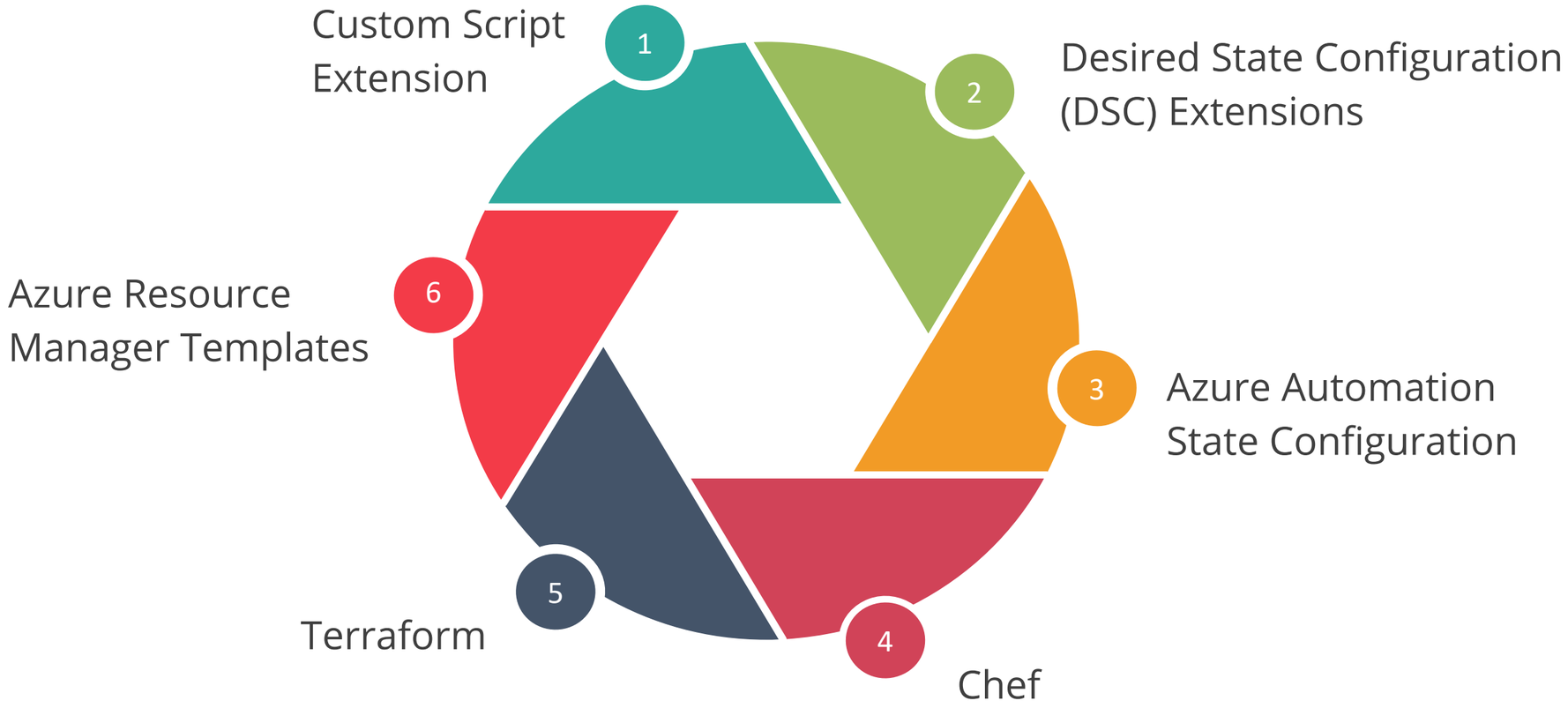
Automated provisioning provides:

Reduced complexity of a deployment configuration by creating a complete architecture using a script or a configuration file.

Deployment in a single operation that automates configuration and accelerates the process.

Automate Infrastructure Deployment

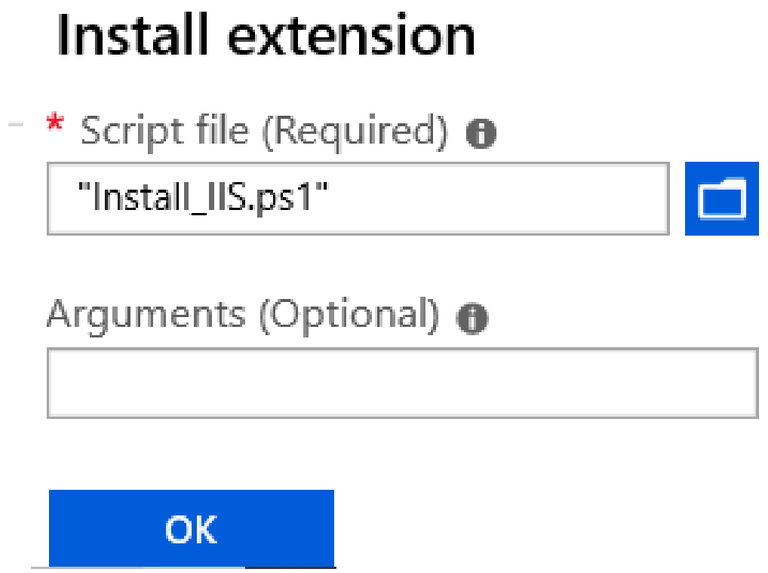
The different services, features, and tools that helps to automate the Infrastructure provisioning are:



Custom Script Extension

The Custom Script Extension tool offers the following features:

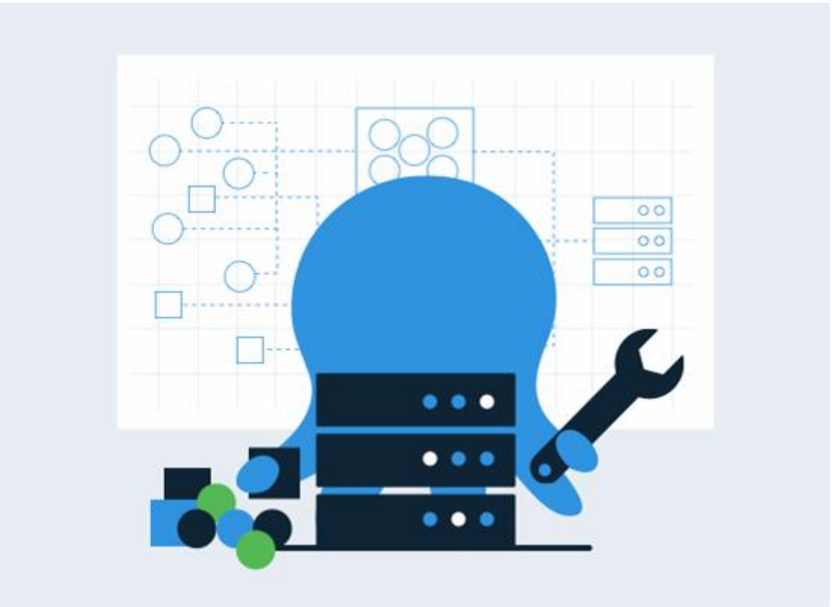
- PowerShell scripts on a file server, GitHub, Azure Storage, or other location can be accessible to a VM
- An extension looks for a script that should be run on the VM
- The script is downloaded and executed on the target VM to apply the changes described in the script
- Add a custom script extension to a virtual machine through Azure Resource Manager templates, PowerShell, or the Azure CLI



Desired State Configuration (DSC) Extensions

DSC defines a state for machines instead of writing manual instructions on how to achieve a state for each machine.

- A DSC extension handler can enforce states
- The configurations for states can be in various locations– Azure Blob storage, internal file storage



Desired State Configuration (DSC) Extensions

- The DSC extension handler grabs the configuration and implements the state on the target VM
- If reboots are necessary, DSC continues to execute the state configuration after the reboots are completed

```
{
        "type": "Microsoft.Compute/virtualMachines/extensions",
        "name": "Microsoft.Powershell.DSC",
        "apiVersion": "2018-06-30",
        "location": "your-region",
        "dependsOn": [
                "[concat('Microsoft.Compute/virtualMachines/', parameters('virtual machineName'))]"
        ],
        "properties": {
                "publisher": "Microsoft.Powershell",
                "type": "DSC",
                "typeHandlerVersion": "2.77",
                "autoUpgradeMinorVersion": true,
                "settings": {
                        "configuration": {
                                "url": "https://demo.blob.core.windows.net/iisinstall.zip",
                                "script": "IisInstall.ps1",
                                "function": "IISInstall"
                        }
                },
                "protectedSettings": {
                        "configurationUrlSasToken": "odLPL/U1p9lvcnp..."
                }
        }
}
```

Desired State Configuration

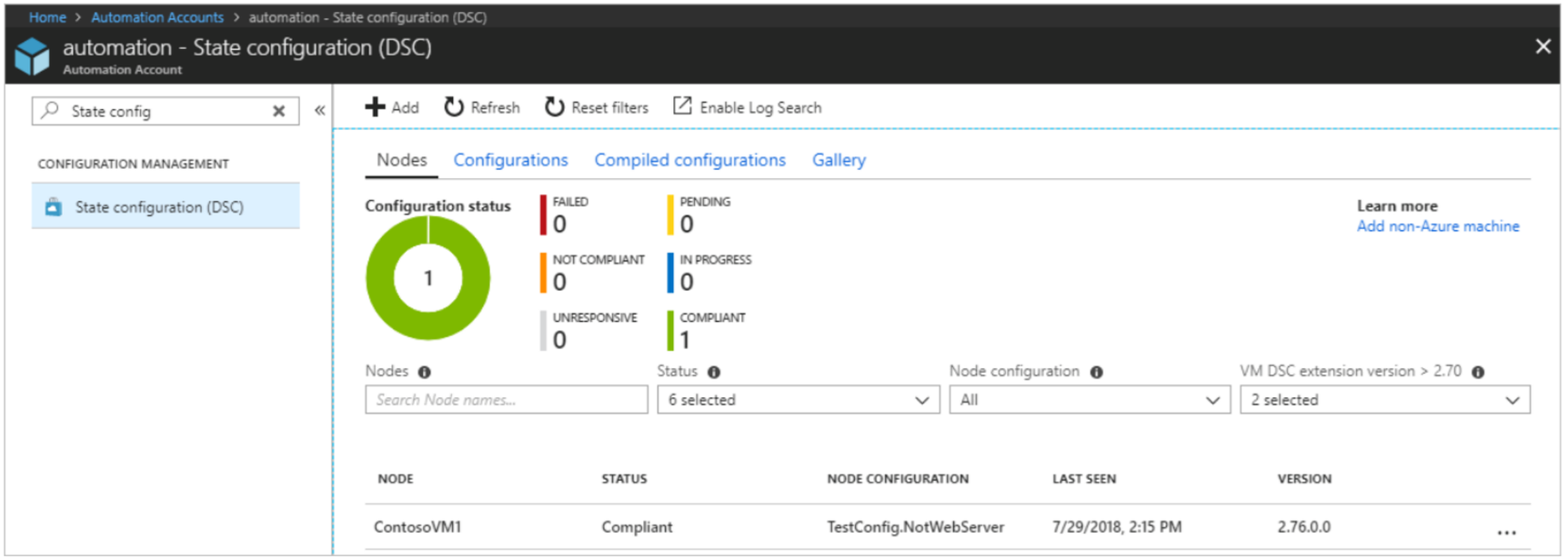
The below mentioned are features of Desired state configuration :

- Configuration block(s) have a name
- Node blocks define the computers or VMs that you are configuring
- Resource block(s) configure the resource and its properties
- There are many built-in configuration resources

```
configuration IISInstall
{
  Node “localhost”
  {
     WindowsFeature IIS
     {
          Ensure = “Present”
          Name = “Web-Server”
      }
   }
}
```

Azure Automation State Configuration

Azure Automation State Configuration service helps in managing and deploying DSC configurations



Source: https://docs.microsoft.com/

Azure Automation State Configuration

These are the ways in which Azure automation state configuration manages a variety of machines.

- Azure VMs, on-premises, and other clouds
- Ensures VMs are assigned the correct configurations automatically
- Each VM reports back on what its current state is and shows whether it has achieved the desired state
- Use Azure portal or Azure PowerShell for implementation

Chef

Chef automates infrastructure deployment, on-premises or in the cloud.

- Chef server can handle 10,000 nodes (machines) at a time
- Chef is hosted and runs as a service
- Chef server is used to manage recipes
- Recipes are commands to run configurations

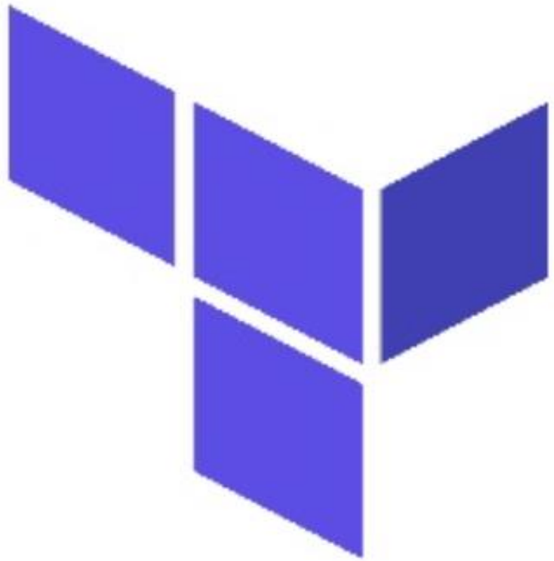
Chef's knife tool is used to deploy VMs and simultaneously apply recipes to VMs

Terraform

Terraform is an open-source infrastructure-as-code software tool.

Terraform Tool

- Executes Terraform scripts
- Uses Hashicorp Configuration Language (HCL) or Json
- Provides easy-to-read script templates, defining resources to create
- Changes can be applied to different cloud service providers (Azure, AWS)
- Requires Terraform installation (VM/CloudShell)



Azure Resource Manager (ARM) Templates

ARM templates are JSON files used to define the Azure resources for deployment.

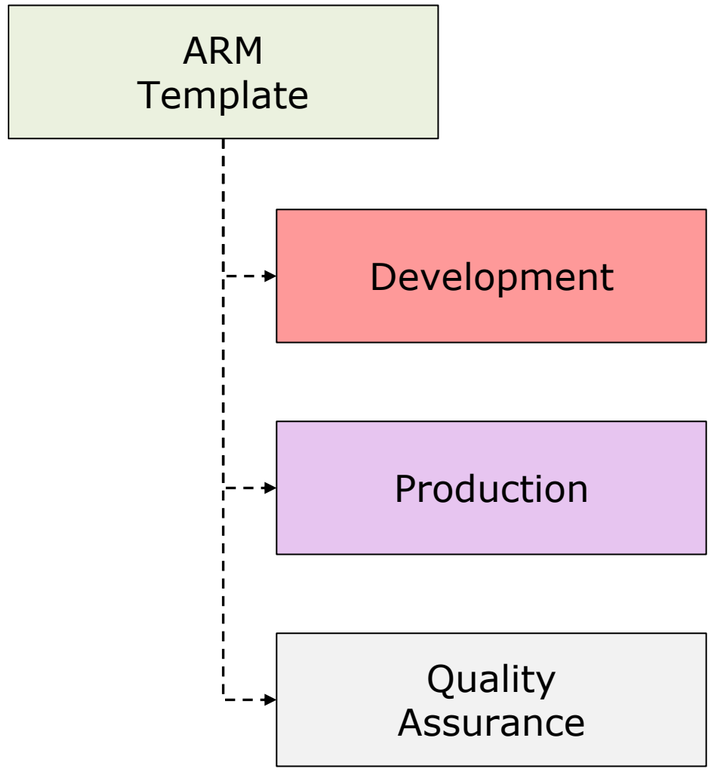
- It enables embedding extensions into VMs
- It helps in deployment using Azure PowerShell, the Azure CLI, or the Azure portal
- It allows testing for ARM templates before deployment

```
"$schema" :
"http://schema.management.azure.com/schemas/2015-01-
01/deployment Template.json#",
"contentVersion": "1.0.0.0",
"parameters": {
"location": {
"type": "string"
},
"storageAccountName": {
"type": "string"
},
"accountType": {
"type": "string"
},
"kind": {
"type": "string"
},
"accessTier": {
"type": "string"
},
"minimumTlsVersion": {
"type": "string"
},
"supportsHttpsTrafficOnly": {
"type": "bool"
},
"allowBlobPublicAccess": {
"type": "bool"
},
```

Template Advantages

Using Resource Manager templates, one can make deployments faster and more repeatable.

- Improves consistency
- Express complex deployments
- Reduce manual, error prone tasks
- Express requirements through code
- Promotes reuse
- Modular and can be linked
- Simplifies orchestration

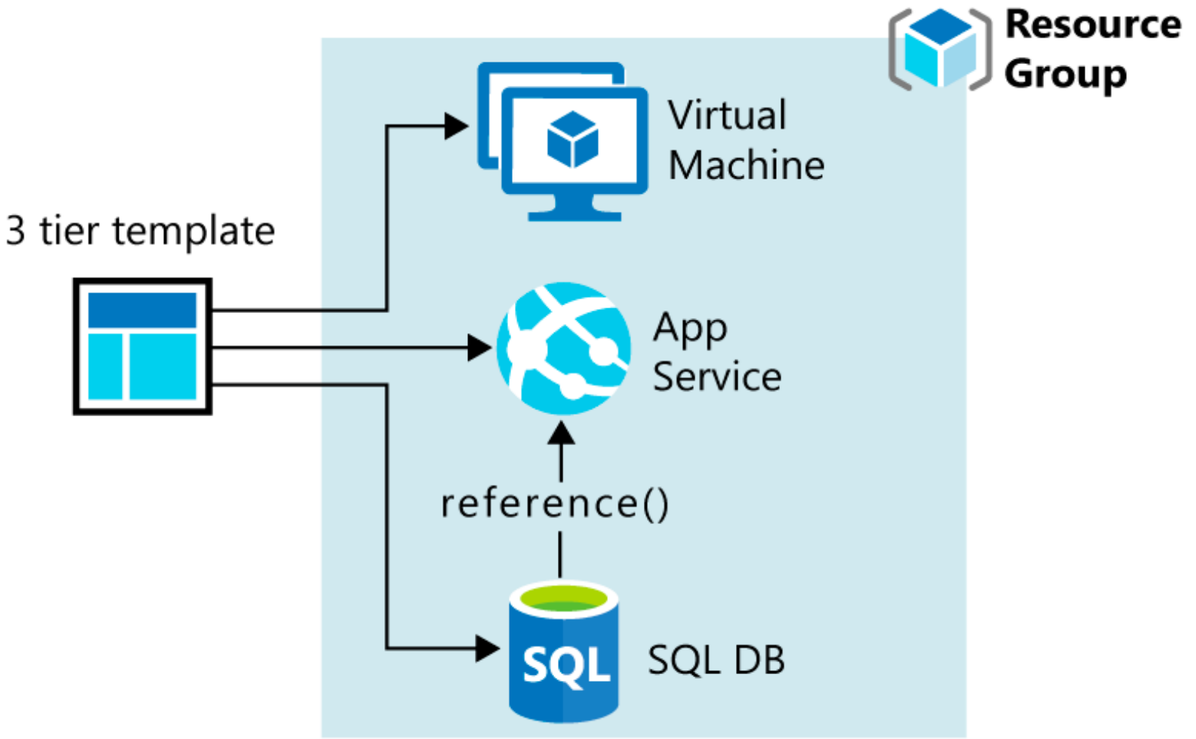


Template Design

Users can define templates and resource groups depending on how you want to manage the solution.

Example

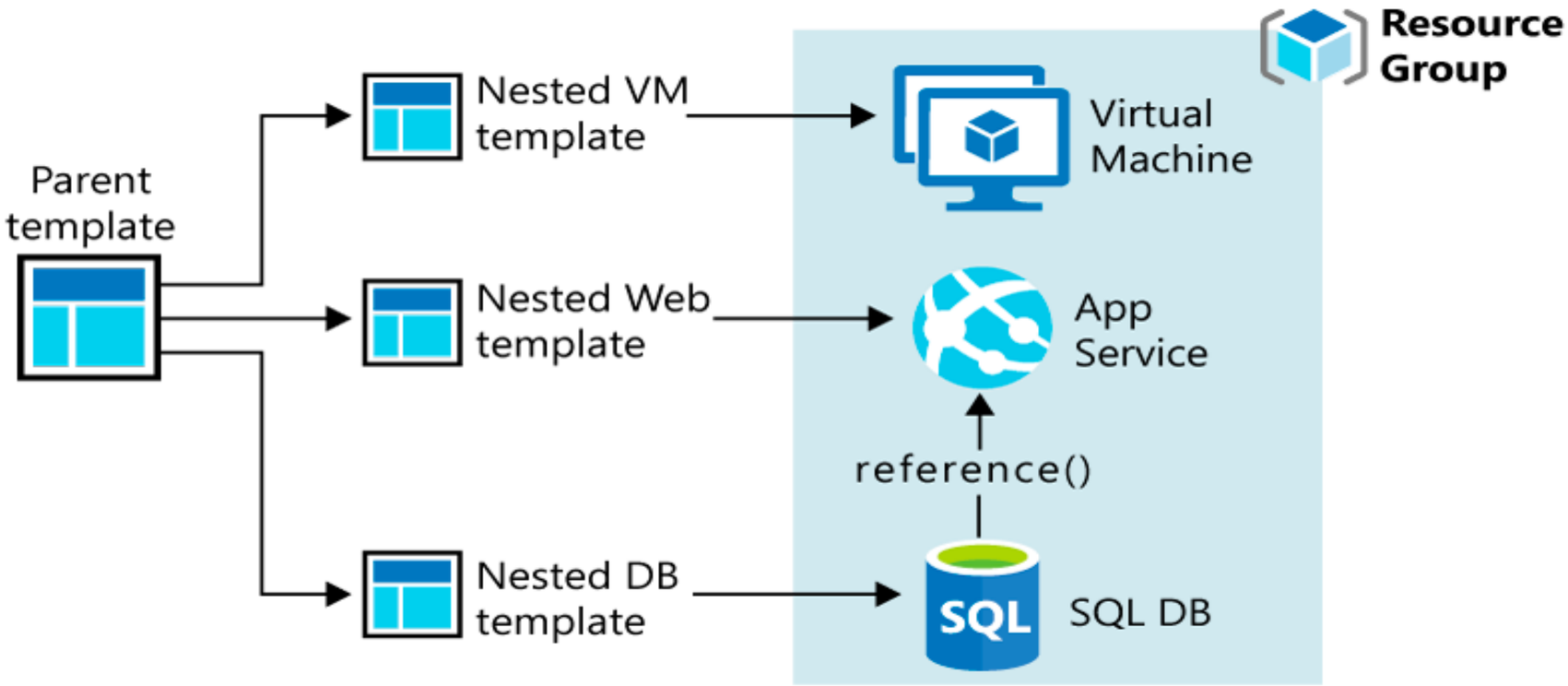
Users can deploy their three-tier application through a single template to a single resource group.



Source: https://docs.microsoft.com/

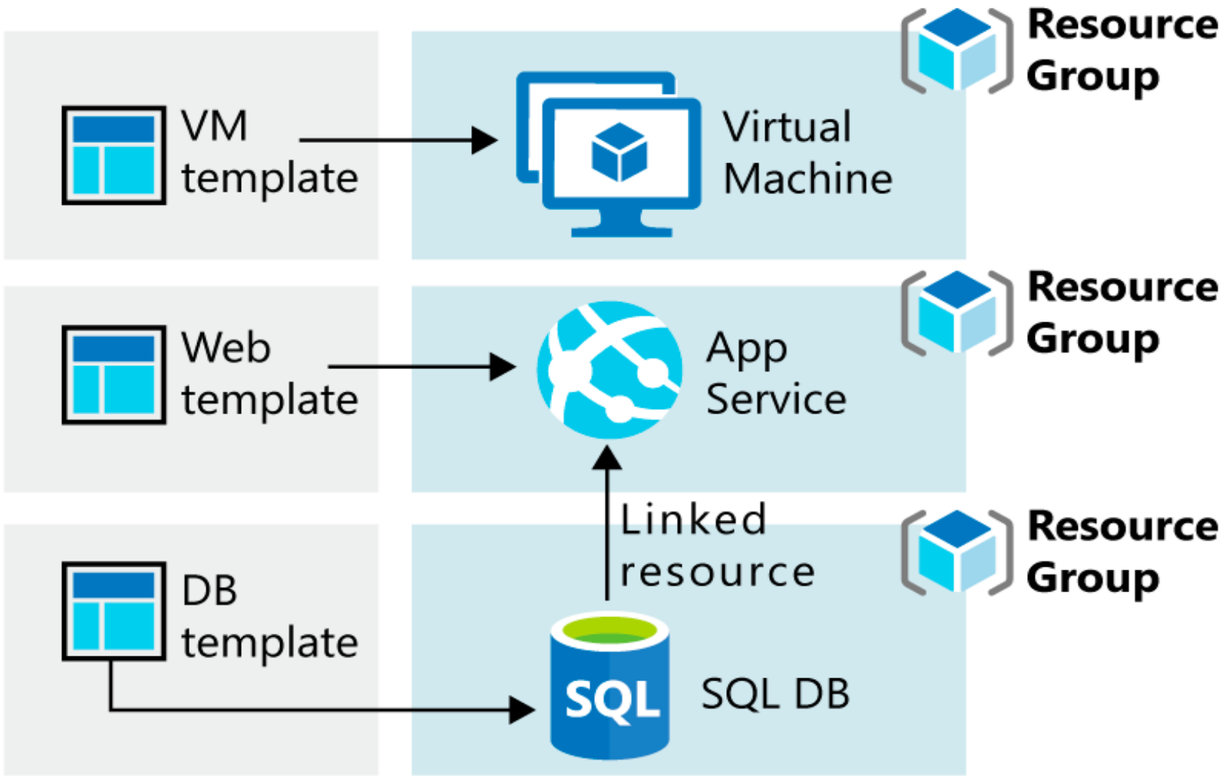
Template Design

Users can divide their deployment requirements into a set of targeted, purpose-specific templates.



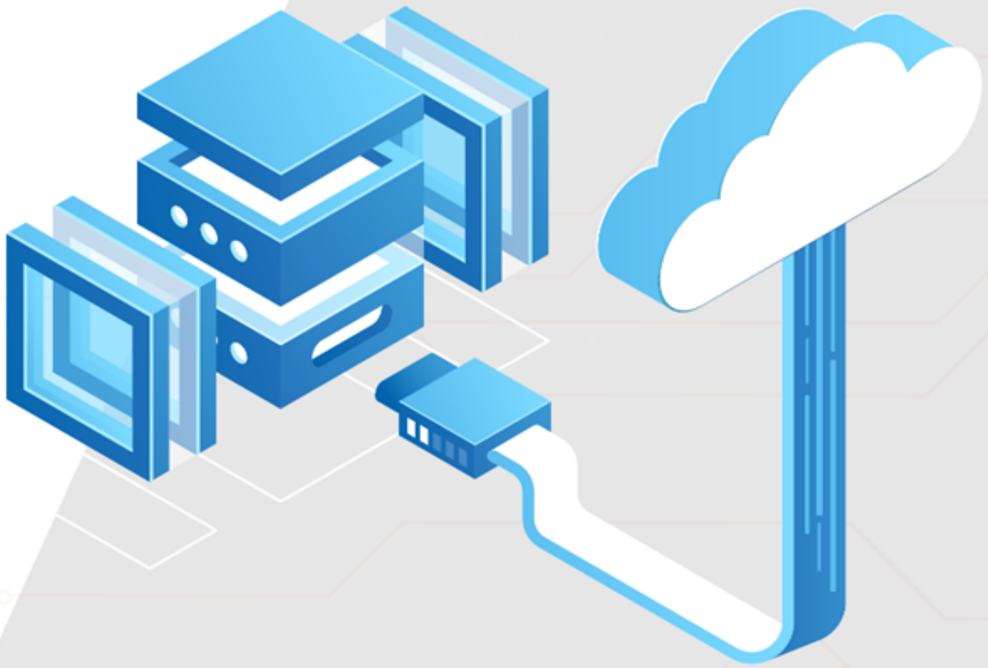
Template Design

If user tiers have separate lifecycles, users can deploy their three tiers to separate resource groups.



Key Takeaways

- AKS is a managed service for running large-scale parallel and high-performance computing (HPC) applications
- Azure Functions is a serverless compute service that supports various function triggers
- A Container can be provisioned with a public IP address and DNS name
- The DSC extension handler grabs the configuration and implements the state on the target VM
- ARM Template enables embedding extensions into VMs



Thank you