

Microsoft Azure Fundamentals Certification and Beyond

Simplified cloud concepts and core Azure fundamentals for absolute beginners to pass the AZ-900 exam



Steve Miles

Foreword by Peter De Tender, Business Program Manager, Enterprise Skills Initiative (ESI) – Microsoft World Wide Learning (WWL), Azure Technical Trainer



Packkt

BIRMINGHAM—MUMBAI

Microsoft Azure Fundamentals Certification and Beyond

Copyright © 2021 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing or its dealers and distributors, will be held liable for any damages caused or alleged to have been caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

Group Product Manager: Vijin Boricha

Publishing Product Manager: Mohd Riyan Khan

Senior Editor: Sangeeta Purkayastha

Content Development Editor: Nihar Kapadia

Technical Editor: Nithik Cheruvakodan

Copy Editor: Safis Editing

Project Coordinator: Shagun Saini

Proofreader: Safis Editing

Indexer: Sejal Dsilva

Production Designer: Nilesh Mohite

First published: January 2021

Production reference: 1241121

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham

B3 2PB, UK.

ISBN 978-1-80107-330-1

www.packt.com

Foreword

Thank you for getting certified in Azure Fundamentals and using this book to help you in your journey. Trust me, this step is the starting point for the rest of your career. Obtaining a technical certification such as AZ-900 will propel your cloud skills and lay the foundation for more Azure certifications to come. I took my first Microsoft exam in 1996 on Windows NT4 Workstation, and I've lost count of how many certifications I have obtained in the last 25 years, but I guess I'm close to 100. Getting certified truly can be a life-changer (hey, it landed me a job at Microsoft!). You put time and effort into learning, deploying, building, and working toward understanding more about a technology. It makes you proud to be part of a worldwide club of Microsoft technology adepts like yourself. It can be a ticket for a pay rise, for a promotion, for moving to a new role... but, most of all, it's a statement to yourself that "*I know this stuff...!!*"

I met Steve several years ago when I was delivering an Azure cloud workshop for UK partners, organized by his company. While at first he hid behind his (pre)sales job title, Steve was actually a lot more technical and also seemed a frequent Microsoft exam-taker. After that joyful week, we stayed in touch to share Azure cloud stories. So, when Steve reached out, asking whether I wanted to be a technical reviewer for his Azure Fundamentals exam prep book, I didn't hesitate a second.

By going through the first chapter, I knew this book was a great read and help for anyone looking to learn about Azure. Besides mapping with the core AZ-900 Microsoft exam objectives, Steve adds a lot of his own experiences, examples, and knowledge into it, as well as closing each chapter with hands-on exercises. That's, to me, the best of both worlds. Whether you learn by reading or by doing, this book will get you on that path to Azure Fundamentals certification, and more. And if you look at the diagrams, know Steve created each of them himself!

If, after reading this book, you have an appetite for more Azure learning content and are thinking of taking more Microsoft exams, make sure you bookmark the

Microsoft Learn website (<https://aka.ms/learn>), with more than 3,000 modules for you to read, watch, or follow along. Azure is a dynamic platform, and so is the certification path. You will love it.

Good luck with your studying journey, and welcome to the club of Microsoft Certified professionals. Let me know when you pass that exam!

Peter De Tender, Business Program Manager, Enterprise Skills Initiative (ESI) – Microsoft World Wide Learning (WWL), Azure Technical Trainer

Contributors

About the author

Steve Miles, aka smiles, has over 20 years of experience in networking, security, data center infrastructure, managed hosting, and cloud solutions.

His experience comes from working for Microsoft's gold partners, including global telco hosters, managed hosting and colocation/data center services providers, global network and public cloud security vendors, hardware, and cloud distribution.

Steve is currently working for a **Cloud Solution Provider (CSP)** indirect distributor based in the UK and Dublin in a cloud and hybrid technology leadership role. He is happiest when in front of a whiteboard, prefers to communicate in illustrations, and is renowned for analogies.

He has over 20 Microsoft certifications, 14 of those being Azure, with many other vendor certifications.

I want to thank the people who have been close to me and supported me, especially my wife, Pippa, aka Mrs Smiles, and my family.

About the reviewers

Tariq Younas is a senior cloud solution architect with more than 15 years of experience in Microsoft technologies, particularly Azure, Azure Stack Hub (Hybrid Cloud), SharePoint, Office 365, DevOps automation, and cloud-native architectures. He has been awarded the Microsoft Azure **Most Valuable Professional (MVP)** award twice.

His areas of interest include public/hybrid cloud, IoT, edge computing, and enterprise architecture. He has successfully worked on application modernization and several IaaS/PaaS migrations. He is an international speaker/author/blogger and a member of IASA Global, an association for all IT architects, and the **Association of Enterprise Architects (AEA)**.

I would like to thank my wife, Sabina Tariq, for her continued support and encouragement with everything that I do. You have always pushed me toward new adventures, accomplishing my goals, and doing what is right. I genuinely appreciate what you have done for us.

Peter De Tender has more than 25 years of experience in the IT industry, with a focus on Microsoft technologies. In early 2012, Peter quickly jumped onto the Azure platform, working as a cloud solution architect and trainer. In mid-2019, Peter took on a position as Azure Technical Trainer within Microsoft, providing Azure Readiness Workshops to large customers and partners within the EMEA region and globally, with a focus on Azure DevOps and Azure architecture.

Peter was an Azure MVP for 6 years, has been a **Microsoft Certified Trainer (MCT)** for more than 12 years, and is still actively involved in the community as a public speaker, technical writer, book author, and publisher.

You can follow Peter on Twitter via @pdttt and check out his technical blog at <http://www.007fflearning.com>.

Ali Khaled Heikal is a Microsoft **Most Valuable Professional (MVP)**. Born and raised in Cairo, Egypt, he graduated from the British University in Egypt with a **Bachelor of Science (B.S.)** in computer science and information systems, top-ranking his specialization. Ali holds many recognitions and professional certifications, and his experience spans multiple technology entities with the world's most reputable organizations. Ali helps organizations with their cloud adoption and optimization strategies to elevate their business success and helps them reap the rewards of digital transformation. He also demonstrates globally recognized technical community leadership. Outside of work, he has an extensive list of hobbies that is forever growing.

To my parents, my two brothers, my friends, and my mentors: thank you for everything.

To Microsoft and the global technical community: thank you for the recognition and continuous support.

Table of Contents

[Preface](#)

Section 1: Cloud Concepts

Chapter 1: Introduction to Cloud Computing.

Where has the cloud computing model evolved from?

Evolution of cloud computing architectures

What is the Shared Responsibility model?

What are the cloud computing delivery models?

Comparing the cloud computing delivery models

What are the cloud computing service models?

A closer look at the cloud computing service models

What is serverless computing?

Comparing the cloud computing service models

Summary

Further reading

Skills check

Chapter 2: Benefits of Cloud Computing.

Why cloud computing?

Cloud computing as a digital transformation enabler

Digital transformation triggers

Migration approach

Cloud computing's target audience

Cloud computing mindset

What is a traditional computing model mindset?

What is a cloud computing mindset?

Cloud computing hierarchy of needs

Cloud computing operations model

Operational benefits of cloud computing.

The economics of cloud computing.

Consumption-based model?

Defining the expenditure models

Applying cost expenditure models to cloud computing.

Thought exercise

Migration assessment discovery output

Migration solution proposal

Summary

Further reading.

Skills check

Section 2: Core Azure Services

[Chapter 3: Core Azure Architectural Components](#)

[Azure global infrastructure](#)

[Azure regions and geographies](#)

[Availability components](#)

[Azure resource management](#)

[Azure management scopes](#)

[Azure management groups](#)

[Azure subscriptions](#)

[Azure Resource Manager](#)

[Hands-on exercises](#)

[Getting started](#)

[Exercise 1 – Azure management groups](#)

[Exercise 2 – Azure access assignment](#)

[Exercise 3 – resource groups](#)

[Exercise 4 – proximity placement groups](#)

[Exercise 5 – availability sets](#)

[Summary](#)

[Further reading](#)

[Skills check](#)

[Chapter 4: Core Azure Resources](#)

[Technical requirements](#)

[Azure resources](#)

[Azure Marketplace](#)

[Azure compute services](#)

[Virtual machines](#)

[Container services](#)

[Azure Container Instances](#)

[Azure Kubernetes Service](#)

[Azure App Service](#)

[The Azure Virtual Desktop service](#)

[Azure network services](#)

[Azure VNets](#)

[Virtual network peering](#)

[Virtual private network gateways](#)

[ExpressRoute](#)

[Azure storage services](#)

[Storage accounts](#)

[Storage tiers](#)

[Storage replication](#)

[Data stores](#)

[Disk Storage](#)

[File storage](#)

[Container \(Blob\) storage](#)

[Azure database services](#)

[Azure SQL Managed Instance](#)

[Azure SQL Database](#)

[Azure Database for MySQL](#)

[Azure Database for PostgreSQL](#)

[Cosmos DB](#)

[Hands-on exercises](#)

[Getting started](#)

[Exercise 1 – creating a VNet](#)

[Exercise 2 – creating a storage account](#)

[Exercise 3 – creating a VM](#)

[Exercise 4 – creating an Azure container instance](#)

[Exercise 5 – creating an Azure web app](#)

[Summary](#)

[Additional information and study references](#)

[Skill check](#)

Section 3: Core Solutions and Management Tools

[Chapter 5: Core Azure Solutions](#)

[Technical requirements](#)

[Serverless computing solutions](#)

[Azure Functions](#)

[Azure Logic Apps](#)

[Artificial intelligence solutions](#)

[Azure Machine Learning](#)

[Azure Cognitive Services](#)

[Azure Bot Service](#)

[Internet of Things solutions](#)

[Azure IoT solutions](#)

[Big data and analytics solutions](#)

[Azure Synapse Analytics](#)

[Azure HDInsight](#)

[Azure Databricks](#)

[DevOps solutions](#)

[The history of DevOps](#)

[What is DevOps?](#)

[Azure DevOps](#)

[GitHub and GitHub Actions](#)

[Azure DevTest Labs](#)

[Thought exercise](#)

Hands-on exercises

Getting started

Exercise 1 – Creating a serverless solution using an Azure Function

Exercise 2 – Creating a serverless solution using an Azure Logic App

Exercise 3 – Creating an IoT solution using an Azure IoT Hub

Exercise 4 – Creating an AI solution using a Bot Service

Summary

Further reading

Skills check

[Chapter 6: Azure Management Tools](#)

[Technical requirements](#)

[Azure portal](#)

[Azure PowerShell](#)

[The Azure CLI](#)

[Azure Cloud Shell](#)

[Terraform on Azure](#)

[Azure mobile app](#)

[Azure Advisor](#)

[Azure Monitor](#)

[Azure Service Health](#)

[Thought exercise](#)

[Hands-on exercise](#)

[Getting started](#)

[Exercise 1 – installing Azure PowerShell](#)

[Exercise 2 – installing the Azure CLI](#)

[Exercise 3 – creating resources using PowerShell from Cloud Shell](#)

[Exercise 4 – creating resources using the Azure CLI from Cloud Shell](#)

[Exercise 5 – exploring Azure Service Health](#)

[Summary](#)

[Further reading](#)

Skill check

Section 4: Security

[Chapter 7: Azure Security](#)

[Technical requirements](#)

[Threat modeling](#)

[Zero Trust](#)

[Defense in depth](#)

[Network and application protection](#)

[NSGs](#)

[Azure Firewall](#)

[Azure DDoS protection](#)

[Azure Key Vault](#)

[Azure Dedicated Host](#)

[Azure Sentinel](#)

[Azure Security Center](#)

[Other protection solutions](#)

[Hands-on exercise](#)

[Getting started](#)

[Exercise 1 – Create an Azure key vault](#)

[Exercise 2 – Secure network access using an NSG](#)

[Summary](#)

[Additional information and study references](#)

[Skills check](#)

Section 5: Identity, Governance, Privacy, and Compliance

[Chapter 8: Azure Identity Services](#)

[Technical requirements](#)

[Azure AD](#)

[Authentication and authorization](#)

[Single sign-on](#)

[MFA and Conditional Access](#)

[Hands-on exercises](#)

[Getting started](#)

[Exercise 1 – creating a new tenant instance of Azure AD](#)

[Exercise 2 – creating users and groups in Azure AD](#)

[Summary](#)

[Further reading](#)

[Skills check](#)

[Chapter 9: Azure Governance](#)

[Technical requirements](#)

[Resource tags](#)

[Resource locks](#)

[Role-based access control](#)

[Azure Policy](#)

[Azure Blueprints](#)

[The Cloud Adoption Framework for Azure](#)

[Hands-on exercises](#)

[Getting started](#)

[Exercise 1 – assigning access with RBAC](#)

[Exercise 2 – creating a custom RBAC role](#)

[Exercise 3 – creating resource locks](#)

[Exercise 4 – enabling resource tagging with Azure Policy](#)

[Exercise 5 – limiting the resource creation location with Azure Policy](#)

[Summary](#)

[Further reading](#)

[Skills check](#)

[Chapter 10: Azure Privacy and Compliance](#)

[Technical requirements](#)

[Core security, privacy, and compliance tenets](#)

[Trust Center](#)

[Microsoft Privacy Statement](#)

[The Product Terms site](#)

[Data Protection Addendum](#)

[Azure compliance documentation](#)

[Azure Sovereign Regions](#)

[Thought exercise](#)

[Hands-on exercise](#)

[Getting started](#)

[Exercise – exploring Microsoft Trust Center Portal](#)

[Summary](#)

[Additional information and study references](#)

[Skills check](#)

Section 6: Cost Management and Service-Level Agreements

[Chapter 11: Azure Cost Planning and Management](#)

[Technical requirements](#)

[Factors that affect costs](#)

[Reducing and controlling costs](#)

[Azure Cost Management](#)

[Azure Pricing calculator](#)

[TCO calculator](#)

[Hands-on exercises](#)

[Getting started](#)

[Exercise 1 – using the Azure Pricing calculator](#)

[Exercise 2 – using the TCO calculator](#)

[Summary](#)

[Further reading](#)

[Skills check](#)

[Chapter 12: Azure Service-Level Agreements](#)

[Technical requirements](#)

[Azure SLAs](#)

[Azure service life cycle](#)

[Thought exercise](#)

[Hands-on exercise](#)

[Getting started](#)

[Exercise 1 – exploring the SLA for a service](#)

[Exercise 2 – exploring Azure Preview features](#)

[Summary](#)

[Additional information and study references](#)

[Skills check](#)

[Chapter 13: Exam Preparation Practice Tests](#)

[Test questions](#)

[Practice test 1 – cloud concepts](#)

[Practice test 2 – core Azure services](#)

[Practice test 3 – core solutions and management tools](#)

[Practice test 4 – security features](#)

[Practice test 5 – identity, governance, privacy, compliance](#)

[Practice test 6 – cost management, SLA, and service life cycle](#)

[Test answers](#)

[Practice test 1 – cloud concepts](#)

[Practice test 2 – core Azure services](#)

[Practice test 3 – core solutions and management tools](#)

[Practice test 4 – security features](#)

[Practice test 5 – identity, governance, privacy, compliance](#)

[Practice test 6 – cost management, SLA, and service life cycle](#)

[Summary](#)

[Additional information and study references](#)

[Other Books You May Enjoy](#)

Preface

Azure is Microsoft's cloud computing platform; it provides organizations on-demand access to compute, storage, networking, and many other resources to meet their needs.

The content of this book is intended to provide complete coverage of the *skills measured* exam requirements to prepare you for the *Microsoft certification exam AZ-900: Microsoft Azure Fundamentals*; it also aims to go beyond the exam objectives, providing an extra depth of knowledge with demonstrable hands-on skills to master, which will be of value in a day-to-day Azure-focused role.

Each chapter provides *thought exercises* and *hands-on exercises* where appropriate to assist in learning the topics covered for every *exam objective's skills measured area*; each chapter then closes with a skills check, further reinforcing the learning of the skills covered.

The book closes with an exam preparation test.

It is important to note that some Microsoft Security Services have been renamed in November 2021. These are renamed as follows:

- **Azure Security Center** and **Azure Defender** are now called **Microsoft Defender for Cloud**
- **Azure Defender plans** to **Microsoft Defender plans**
- **Azure Sentinel** is now called **Microsoft Sentinel**
- **Azure Defender for IoT** is now called **Microsoft Defender for IoT**
- **Azure Defender for SQL** is now called **Microsoft Defender for SQL**
- **Microsoft Cloud App Security** is now called **Microsoft Defender for Cloud App**
- **Microsoft Defender for Business** is introduced as a new Service SKU

You can learn more about the update to Microsoft security services at this url:

<https://docs.microsoft.com/en-us/azure/defender-for-cloud/defender-for-cloud-introduction>

Who this book is for

This book is for those in commercial, operational, or technical roles looking to pass the *Microsoft certification exam AZ-900: Microsoft Azure Fundamentals*. The exam is intended for candidates who are just beginning to work with cloud-based solutions and services or are new to Azure. Anyone who already has the certification and is looking to see the hands-on practical side of the Azure services can also benefit from this book.

What this book covers

[Chapter 1](#), *Introduction to Cloud Computing*, defines cloud computing and describes public cloud, private cloud, and hybrid cloud; compares and contrasts the three types of cloud computing; describes **Infrastructure-as-a-Service (IaaS)**, **Platform-as-a-Service (PaaS)**, **Software-as-a-Service (SaaS)**, and serverless|**Functions-as-a-Service (FaaS)**; and identifies a service type based on a use case.

[Chapter 2](#), *Benefits of Cloud Computing*, identifies the benefits of cloud computing and Microsoft Azure; describes scalability, elasticity, agility, high availability, and disaster recovery; describes the consumption-based model; and identifies the differences between **capital expenditure (CapEx)** and **operational expenditure (OpEx)**.

[Chapter 3](#), *Core Azure Architectural Components*, describes the core architectural physical components, such as data centers, networks, Regions, Availability Sets, and Availability Zones, and describes the core architectural, logical components, such as subscriptions, management groups, resource manager, and resource groups.

[Chapter 4](#), *Core Azure Resources*, explains Azure resources; describes the benefits and usage of Azure Marketplace; describes the benefits and usage of compute services such as Azure Virtual Machines, Azure App Service, **Azure Container Instances (ACI)**, **Azure Kubernetes Service (AKS)**, Logic Apps/Functions, and Virtual Desktop; describes the benefits and usage of storage services such as blob storage, disk storage, file storage, and storage tiers; describes the benefits and usage of network services such as Virtual Networks, VPN Gateway, virtual network peering, and ExpressRoute; and describes the benefits and usage of database services such as Cosmos DB, Azure SQL Database, Azure Database for MySQL, Azure Database for PostgreSQL, and SQL managed instances.

[Chapter 5](#), *Core Azure Solutions*, describes serverless computing solutions, including Azure Functions and Logic Apps, Azure Machine Learning, Cognitive Services, Azure Bot Service, **Internet of Things (IoT)** Hub, IoT Central, Azure Sphere, Azure Synapse Analytics, HDInsight, Azure Databricks, Azure DevOps, GitHub, GitHub Actions, and Azure DevTest Labs.

[Chapter 6](#), *Azure Management Tools*, describes the functionality and usage of the Azure portal, Azure PowerShell, Azure CLI, Cloud Shell, the Azure mobile app, Azure Advisor, Azure Monitor, and Azure Service Health.

[Chapter 7](#), *Azure Security*, describes threat modeling, the concept of defense in depth; the functionality and usage of Key Vault and Azure dedicated hosts; the functionality and usage of **network security groups (NSGs)**, Azure Firewall, and Azure DDoS Protection; and the functionality and usage of Azure Security Center and Azure Sentinel.

[Chapter 8](#), *Azure Identity Services*, defines Azure Active Directory; describes the functionality and usage of Azure Active Directory and explains the difference between authentication and authorization; and describes the functionality and usage of Conditional Access, **multi-factor authentication (MFA)**, and **single sign-on (SSO)**.

[Chapter 9](#), *Azure Governance*, describes the functionality and usage of resource tags, resource locks, role-based access control, Azure Policy, and Blueprints, and describes the Cloud Adoption Framework for Azure.

[Chapter 10](#), *Azure Privacy and Compliance*, describes the core Microsoft tenets of security, privacy, and compliance, and describes the purpose of the Trust Center, Microsoft Privacy Statement, the Product Terms site, **Data Protection Addendum (DPA)**, Azure compliance documentation, Azure Sovereign Regions, Azure Government cloud services, and Azure China cloud services.

[Chapter 11](#), *Azure Cost Planning and Management*, covers factors impacting the cost of resources and options for cost control and reduction and explains Azure Cost Management. It also covers the Azure pricing calculator and the **Total Cost of Ownership (TCO)** calculator.

[Chapter 12](#), *Azure Service-Level Agreements*, describes the purpose of an Azure **service-level agreement (SLA)** and identifies actions that can impact an SLA, positively or negatively, such as Availability Zones or composite SLAs. It also describes the service life cycle in Azure, such as development, private and public preview, and **general availability (GA)**.

[Chapter 13](#), *Exam Preparation Practice Tests*, provides exam preparation tests for key skills measured from the exam objectives.

To get the most out of this book

This book's content is intended for candidates who are just beginning to work with cloud-based solutions and services or are new to Azure; no specific knowledge is assumed or required.

To carry out the exercises in this book, you will require the following:

- Access to an internet browser.
- A Microsoft account. If you do not have a Microsoft account, you can create a free account at <https://account.microsoft.com/account>.
- An Azure subscription that has access to create and delete resources in the subscription. If you do not have an Azure subscription, you can create a free Azure account at <https://azure.microsoft.com/free>.
- You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.

Software/hardware covered in the book	Operating system requirements
PowerShell	Windows, macOS, or Linux
Azure CLI	Windows, macOS, or Linux
Azure Cloud Shell	Windows, macOS, or Linux

Download the color images

We also provide a PDF file that has color images of the screenshots and diagrams used in this book. You can download it here: https://static.packt-cdn.com/downloads/9781801073301_ColorImages.pdf.

Conventions used

There are a number of text conventions used throughout this book.

`code in text`: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "In the search bar, type `management groups`."

Any command-line input or output is written as follows:

```
$PSVersionTable.PSVersion
```

Bold: Indicates a new term, an important word, or words that you see onscreen. For instance, words in menus or dialog boxes appear in **bold**. Here is an example: "Click on **Management groups** from the list of services shown."

TIPS OR IMPORTANT NOTES

Appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: If you have questions about any aspect of this book, email us at customercare@packtpub.com and mention the book title in the subject of your message.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit www.packtpub.com/support/errata and fill in the form.

Piracy: If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at copyright@packt.com with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit authors.packtpub.com.

Share Your Thoughts

Once you've read *Microsoft Azure Fundamentals Certification and Beyond*, we'd love to hear your thoughts! Please [click here to go straight to the Amazon review page](#) for this book and share your feedback.

Your review is important to us and the tech community and will help us make sure we're delivering excellent quality content.

Section 1: Cloud Concepts

This section provides complete coverage of the knowledge and skills required for the *skills measured* of the *Describe cloud Azure concepts* section of the exam. We also cover knowledge and skills that go beyond the exam content, so you are prepared for a real-world, day-to-day Azure-focused role.

This part of the book comprises the following chapters:

- [Chapter 1](#), *Introduction to Cloud Computing*
- [Chapter 2](#), *Benefits of Cloud Computing*

Chapter 1: Introduction to Cloud Computing

Cloud computing is somewhat of a misnomer; it's a marketing term for a technology model that an organization may adopt to consume computing resources. It can benefit many audiences, addressing a need to access on-demand computing resources that are self-service, automated, and elastic to cater to demand.

Its value to a business is as an enabler for digital transformation and innovation, quicker time to market and value, economies of scale, a flexible cost model, and an agile operating model; this ability gives a choice in how computing resources can be provided and consumed by a business to suit their operating model in the most appropriate way.

Microsoft, Amazon, and Google are some of the public cloud computing platform providers. These providers own hardware on their facilities, from which they create computing resources that are made available to all tenants on the platform, which use their portion of the resources and get billed only for what they use. The users of these computing resources benefit through what is described as economies of scale.

The objectives for this chapter cover the AZ-900 Azure Fundamentals exam skills area *Describe Cloud Concepts*.

By the end of this chapter, you will have learned the skills to be able to do the following:

- Define cloud computing.
- Describe public cloud, private cloud, and hybrid cloud.
- Compare and contrast the three types of cloud computing.
- Describe **Infrastructure as a Service (IaaS)**, **Platform as a Service (PaaS)**, **Software as a Service (SaaS)**, and **Serverless | Functions as a Service (FaaS)**.
- Identify a service type based on a use case.

In addition, this chapter's goal is also to take your knowledge beyond the exam content so you are prepared for a real-world, day-to-day Azure-focused role.

It is important to note that in November 21 some Microsoft Security Services have been renamed. These are renamed as follows:

- **Azure Security Center** and **Azure Defender** are now called **Microsoft Defender for Cloud**
- **Azure Defender plans** to **Microsoft Defender plans**
- **Azure Sentinel** is now called **Microsoft Sentinel**
- **Azure Defender for IoT** is now called **Microsoft Defender for IoT**
- **Azure Defender for SQL** is now called **Microsoft Defender for SQL**
- **Microsoft Cloud App Security** is now called **Microsoft Defender for Cloud App**
- **Microsoft Defender for Business** is introduced as a new Service SKU

You can learn more about the update of Microsoft security services at this url:

<https://docs.microsoft.com/en-us/azure/defender-for-cloud/defender-for-cloud-introduction>

Where has the cloud computing model evolved from?

Cloud computing is the next phase in the evolution of the computing platform and the next significant shift of the IT industry; it is another model for delivering computing resources to an organization.

We have evolved from physical hardware to virtualization that is run from our facilities or somebody else's to another computing model shift of virtual machines to containers and now the leap to serverless, where the business logic layer is the new scale unit in the journey to the cloud:

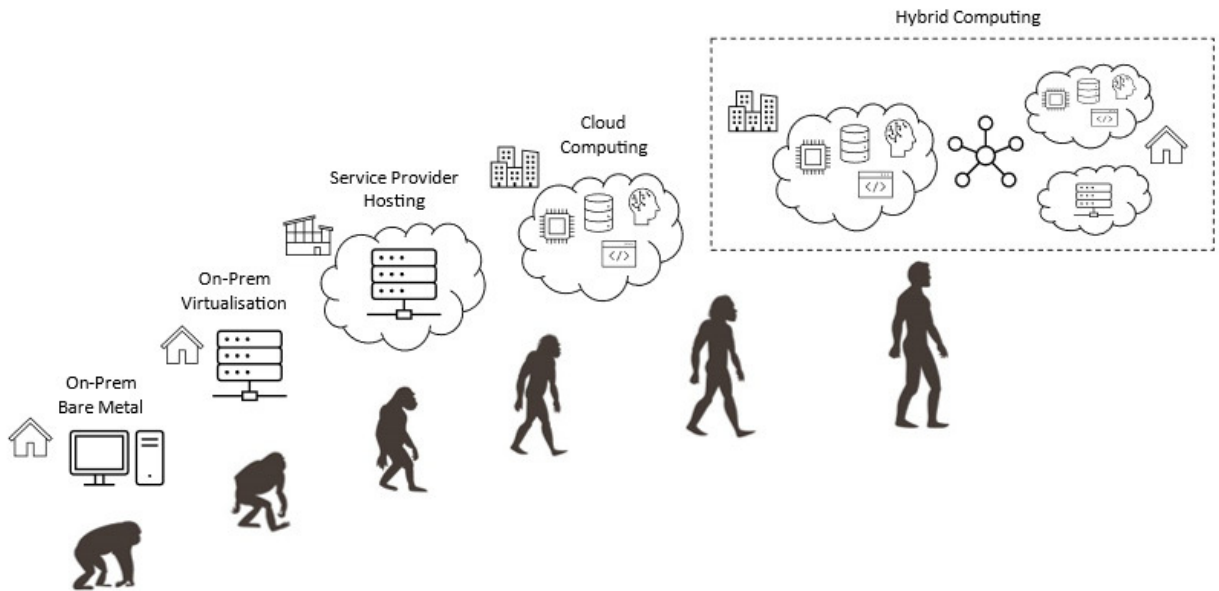


Figure 1.1 – Evolution of cloud computing

The goal of hybrid computing is to provide *computing resources anywhere, anytime*, and give a business the power of choice as to the most suitable technology platform for any given workload, business initiative, or scenario that needs to be supported.

Cloud computing is not only a technical evolution but also a financial evolution; the expenditure model shifts from that of **capital expenditure (CapEx)** of hardware (buying upfront before you can use resources) to **operating expense (OpEx)** and paying as you use resources.

It should be noted, though, that the private cloud model can contain an element of *CapEx* and *OpEx*; typically (*and for the exam objectives*), the primary cost expenditure model is *CapEx*. However, leased hardware and software are financially also considered *OpEx*, but would mainly mean building an on-premises infrastructure.

As the computing platform environments have changed over time, so have the architectures; this next section will look at the evolution of the cloud computing architectures.

Evolution of cloud computing architectures

Serverless comes about from another architectural shift in the compute layer and is an extension and evolution of PaaS.

When you use PaaS resources to host a website or application or execute code, you are still using servers; you specify a set of underlying compute resources and pay for those. This would be the server farm in traditional hosting.

Whereas in serverless, it's exactly as it says in the name, you are not responsible for creating any compute resources; there are servers involved, but this compute layer is provided by the platform provider – it's abstracted from your control or responsibility. In essence, you provide your business logic layer, and they run it for you on their compute layer.

The term **cloud-native** also gets introduced here; this means moving from the *monolith stacks of virtual machines* to *microservices* such as *containers* or *serverless* architecture solutions as *functions* (Azure Functions) or *workflows* (Azure Logic Apps). This is a fundamental shift from *compute stack-centric* to *business logic-centric*, where we are only focusing on the outcomes and not the inputs; that is, we no longer care or have to concern ourselves with the lower layers such as the languages, runtimes, compute, and so on, as these are now provided as a service for us to consume by the provider. You give the code, and the provider will decide how they will handle the execution of it.

As I mentioned earlier, serverless is about abstracting the *language runtime*, PaaS is about abstracting the *compute*, and IaaS is about abstracting the *hardware*.

When we say *abstract*, what we mean is *to remove*, that is, remove the requirement to provide that layer; we make that layer the cloud provider's responsibility to provide, scale, keep available, maintain, and so on. It is a layer that we no longer need to know or care about:

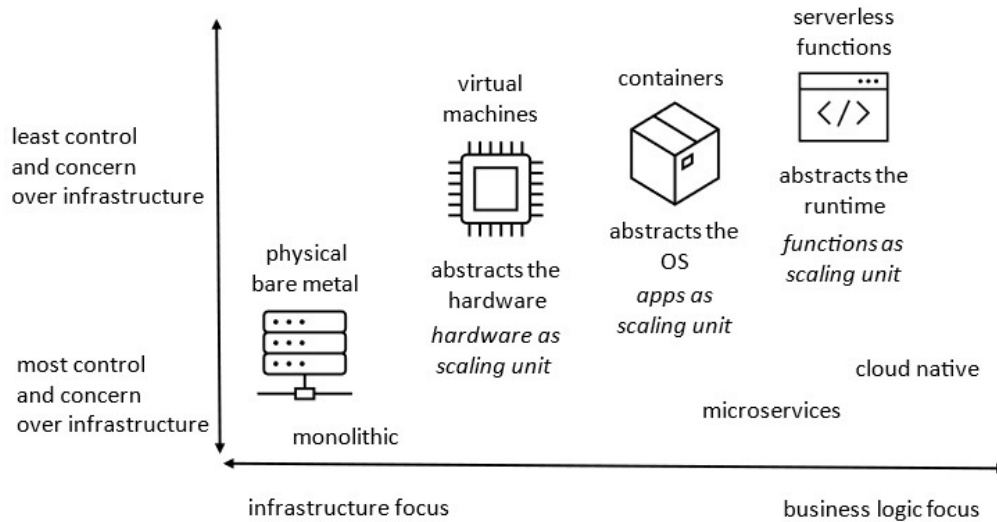


Figure 1.2 – Cloud computing architectures

This section looked at the evolution of both the computing platform environments and the cloud computing architectures. The illustration outlines where the architectures differ in their characteristics and outlines decision criteria to consider so that each architecture can be positioned to make the most appropriate choice for any given scenario.

In the next section, we look at the *Shared Responsibility Model*, one of the most misunderstood cloud computing concepts but one of the most critical to understand. It underpins many decisions and their consequences of security and degrees of control measures.

In the *Comparing the cloud computing service models* section, we continue to look at the degrees of control offered by each model.

What is the Shared Responsibility model?

The **Shared Responsibility model** is a security model and is critically important to understand when operating resources within a public or hybrid cloud environment.

You should understand when it is *your* responsibility to provide the appropriate level of security and where it's *not* your responsibility but that of the cloud services provider.

This responsibility level may dictate what cloud computing services models you decide to deploy, such as IaaS, PaaS, or SaaS, to determine how much control and responsibility you must provide or hand off to the cloud services provider.

There are three levels of responsibility to be considered:

- Responsibilities that the consumer of the cloud services *always retains*
- Responsibilities that can vary by the *resource type*
- Responsibilities that will *transfer* to the cloud services provider:

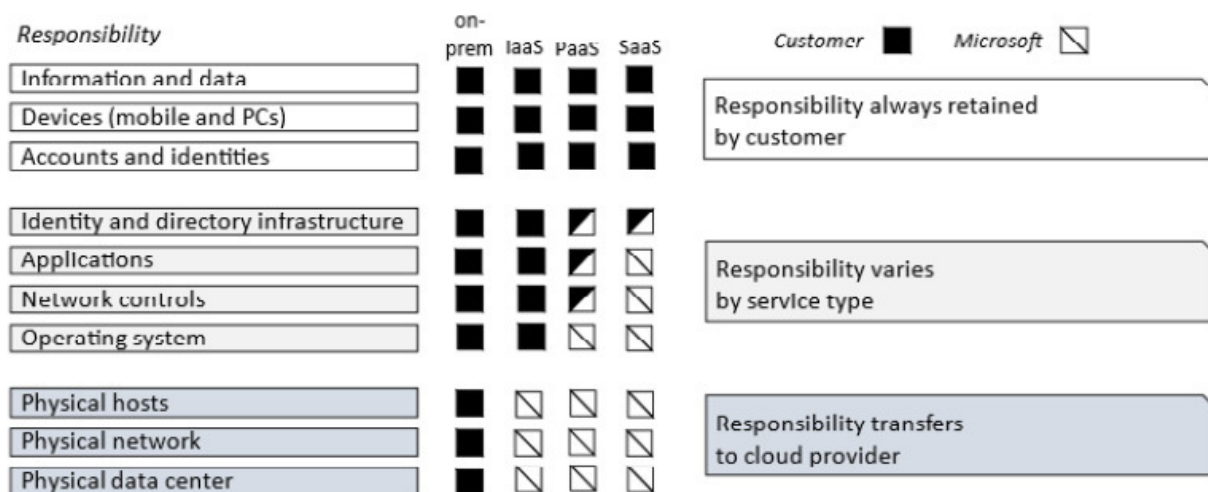


Figure 1.3 – Shared Responsibility model

This security model illustration aims to visually set out the division or separation of responsibilities between the consumer of the cloud resources and the cloud services provider itself.

The most critical to be aware of is the responsibilities that the consumer of cloud services *always retains* and *your responsibilities* to secure and protect.

What are the cloud computing delivery models?

Cloud computing generally has three deployment models: **public cloud**, **private cloud**, and **hybrid cloud**:

- **Public cloud**, in a nutshell, is a *shared entity (multi-tenant)* computing model. Hardware and resources such as compute, storage, and networking are owned by the cloud provider and shared with other tenants on the platform, known as *multi-tenant* or *multi-tenancy*. Think of this as *an apartment block*, where you are a tenant that shares the building with other tenants; you pay rent to a landlord for your apartment. In cloud computing, this is the *service provider*.
- **Private cloud**, in a nutshell, is a *dedicated entity (single-tenant)* computing model. Hardware and resources such as compute, storage, and networking are dedicated to your organization use only; this is *single-tenant*. Think of this as a *house* as opposed to an apartment block; you are the *single tenant*, and you *do not share* the building with any other tenants. You either own the building or you rent the property and pay a landlord; that is, a private cloud can be hardware that you own in your facility or a third-party hosting provider, colocation data center facilities provider. Alternatively, this could be their hardware that they dedicate to you, which is traditional dedicated server hosting.
- **Hybrid cloud**, in a nutshell, is a combination of a *shared entity (multi-tenant)* computing model and a *dedicated entity (single-tenant)* computing model. Some computing resources you choose to have running in your private cloud environment and some resources you choose to have running in a public cloud environment based on your needs. This model offers the most agility and flexibility to changes in demand and business requirements:

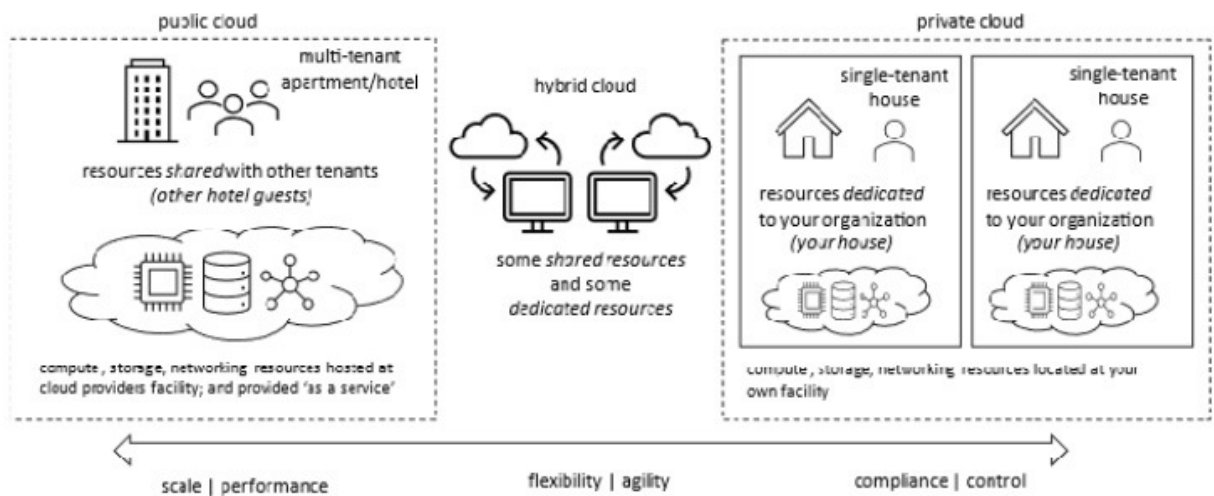


Figure 1.4 – Cloud computing delivery models

This illustration aims to outline some key aspects of the three delivery models of *public*, *private*, and *hybrid* cloud.

In the following section, we will compare each of these delivery models and look at the characteristics of each model in more detail.

Comparing the cloud computing delivery models

From the last section, we can now define what the delivery models are. This section looks at the characteristics of each model in more detail to help you understand when you may choose one over the other.

Each delivery model has several characteristics. The most appropriate model is defined by how much you want (or need/have mandated) to control, secure, and manage your resources, for example, your apps, code, data, networks, security, and so on.

The deployment model defines what control you have over your cloud computing resources, for example, your apps, data, networks, security, and so on. It describes what resources you share or have dedicated for your organization's use.

We use the terms *multi-tenant* and *single-tenant* to differentiate between models that share resources or have dedicated resources.

We could analogize this to a house versus a hotel; with a house, you have your private and dedicated front door, stairs, kitchen, TV/movie subscription service, and more, whereas with a hotel, you have a private room dedicated to you for your sole use, but you share a front door, stairs, kitchen/restaurant, TV/movie subscription service, and so on:

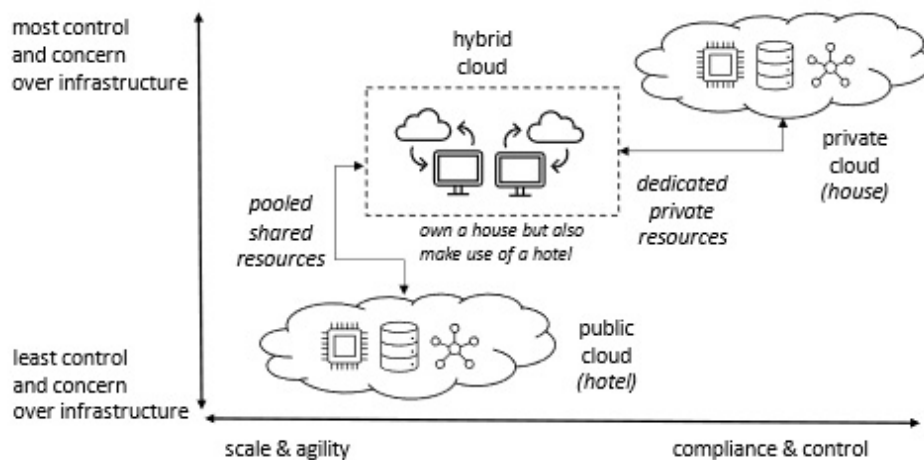


Figure 1.5 – Comparing cloud computing delivery models

Now that we have a basic understanding of the delivery models, this next section will cover the characteristics of each delivery model in more depth.

Characteristics of public cloud computing resources

To recap, a public cloud is a *shared entity* (multi-tenant) computing model.

The following are the characteristics of public cloud computing resources:

- Metered pricing and consumption-based billing and pay-as-you-go monthly usage costs; you only pay for the resources you use, which can allow cost control and cost management.
- Almost unlimited resources are available.
- Performance, scalability, and elasticity. Rapid, on-demand, and automated provisioning and de-provisioning computing resources are required.
- Availability, reliability, fault tolerance, and redundancy.
- Computing resources access is available anywhere, typically via the internet and a private managed network such as Microsoft's ExpressRoute service.
- Self-service management, typically through a web browser or a command-line interface.
- Least control over security, protection, and compliance; you *do not* have complete control over security and compliance with the public cloud model.
- Access to computing resources can be provided by Azure Active Directory as the identity and authentication layer and traditional Windows Server Active Directory when you synchronize the directories.
- Physical hardware is *not/cannot* be deployed to public cloud computing platforms; virtual servers are provided. However, some cloud providers allow physical hardware to be dedicated to an organization's use.
- May allow on-premises facilities hosting computing resources to be decommissioned.
- Expenditure model; move from a CapEx model to an OpEx model. No CapEx on hardware.

The following *giants* are use case examples of public cloud platforms: Microsoft Azure, **Amazon Web Services (AWS)**, and **Google Cloud Platform (GCP)**.

Characteristics of private cloud computing resources

To recap, a private cloud is a *dedicated entity* (single-tenant) computing model.

The following are the characteristics of private cloud computing resources:

- Computing resources created on-premises at the organization's facility or could be provided at a third party's hosting facility; resources only available within the capacity provisioned.
- It requires a CapEx expenditure model for computing resources.
- Computing hardware (physical servers/virtualization platforms and so on) is implemented for the organization's sole use. The hardware/physical resources must be supported; failed hardware must be replaced.
- Required to provide systems and data availability, fault tolerance, scalability, security, protection, update management, maintenance, and support.
- May allow on-premises facilities hosting computing resources to be decommissioned.
- Computing resources access is available via a local/private network and typically will have an internet connection. The private cloud resources, however, may be disconnected from the internet or have intermittent access in scenarios such as cruise ships, construction sites, and Formula One teams on the trackside; while some other scenarios, such as regulated or high-security facilities such as medical, research, scientific, defense, and manufacturing, may not permit internet access and so are disconnected from the internet. Being connected or disconnected from the internet is not a defining characteristic of private clouds.
- The same self-service management functionality and creation of resources is provided as with the public cloud computing model, but you remain in complete control of the security and governance; and you are also entirely responsible for the purchase, implementation, maintenance, and support of the hardware and computing resources you provide from the private cloud platform.
- You *do* have complete control over hardware, physical resources, security, and compliance with the private cloud model.
- Traditional Windows Server Active Directory can provide access to computing resources as the primary identity and authentication layer; Azure Active Directory can also be utilized when connecting to public cloud computing resources through a hybrid model by using directory synchronization as the link between the two identity providers for a consistent, common, or same-sign-on experience.
- Physical servers *can* be deployed with the private cloud model.

The following are examples of private cloud platforms: Azure Stack or VMware VCloud.

Characteristics of hybrid cloud computing resources

To recap, a hybrid cloud is a combination of a *shared entity* (multi-tenant) computing model and a *dedicated entity* (single-tenant) computing model.

The following are the characteristics of hybrid cloud computing resources:

- The greatest flexibility in choosing the most appropriate location of computing resources and computing model.
- The hybrid cloud model provides a choice of creating some computing resources created in the service providers' public cloud computing platforms; some resources are created in your on-premises private cloud platform; both these resources are connected via the internet or a private managed network such as Microsoft's ExpressRoute service.
- It allows bursting or extend computing resource capacity to a public cloud.
- Computing hardware (physical servers/virtualization platforms and so on) is implemented for the organization's sole use as part of the private cloud resources. These hardware/physical resources must be supported; failed hardware must be replaced. For public cloud resources, the hardware and physical resources are provided and supported by the service provider of the public cloud resources.
- It provides the greatest flexibility of access to computing resources via the internet or private networks.
- Private clouds are not necessarily disconnected from public cloud resources; access may be provided by a private managed network such as ExpressRoute to allow a hybrid cloud approach, a computing model where an organization uses some public cloud resources connected to some private cloud resources.
- It provides the greatest flexibility of control of security, protection, and compliance.
- Traditional Windows Server Active Directory can provide access to computing resources as the primary identity and authentication layer; Azure Active Directory can also be utilized when connecting to public cloud computing resources through a hybrid model by using directory synchronization as the link between the two identity providers for a consistent, common, or single-sign-on experience.
- Physical servers *can* be deployed within the private cloud and public cloud, but you *cannot own* these servers in the public cloud; they can only be *rented*.
- It provides the greatest flexibility of expenditure model, that is, the ability to choose CapEx or OpEx, whichever is most appropriate for the computing resources.

The following is an example of a hybrid cloud platform: Azure Stack connected to Azure – this scenario could have on-premises virtual machines backing up to Azure or an Azure web app connecting to an on-premises SQL Server, for example.

In this section, we saw the different cloud computing *delivery models*, how they compare, and the characteristics of each. Now we will take the same approach to look at the cloud computing *service models*.

What are the cloud computing service models?

Cloud computing has four service models; these could also be referred to as categories. These service models are as follows:

- IaaS
- PaaS
- Serverless/FaaS
- SaaS

The following illustration shows the relationship between these service models. Every cloud computing resource will fit into one of these three categories; a solution will comprise one or more resources from each of these categories, and you would select the resources from each category based on each solution's needs, a mix and match approach to tailoring a set of technology resources to map to business needs:

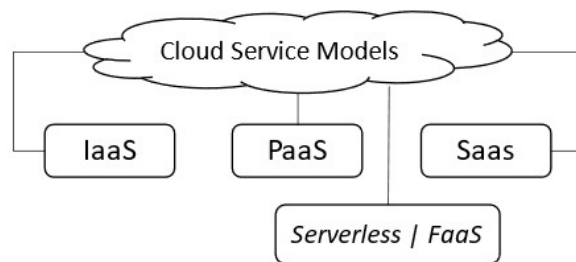


Figure 1.6 – Cloud computing service models

In this next section, we will cover quite a lot of ground as there are many concepts and aspects on which to present information, not only for the exam but also for knowledge beyond. We will take a closer look at each of the service models, comparing and contrasting each of their characteristics in more detail and introducing the concept of *serverless* computing.

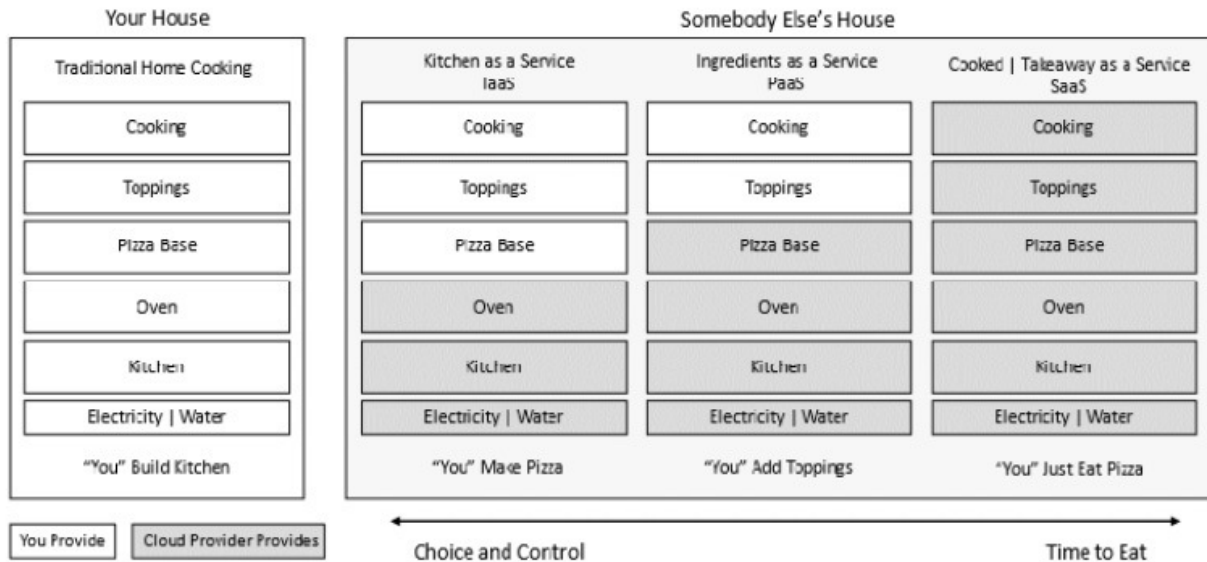


Figure 1.8 – Cloud computing service models – layer analogy

The comparison in this illustration to the cloud computing service model is to what level of input and control you want to fulfill that requirement to eat some pizza.

Do you want the quickest time to be enjoying eating pizza and let somebody else take care of making it, selecting the toppings, and cooking it? The trade-off is that you don't get a say in what ingredients they use to make the pizza base, what size it is, what toppings, or how well it is cooked; but in this scenario, you accept that you don't need to know or care and this may be the perfect scenario to meet this particular need at this time. This is an example of *SaaS*, a ready-cooked and prepared meal ready to consume, that is, *takeaway as a service*.

However, to contrast the example, you may need to eat pizza the same as before, but you have some requirements you wish to specify or mandate. You want a particular type of flour or ingredient to be used in the pizza base; maybe this is imperative as you have a specific intolerance or medical condition, so you must have control over the ingredients. You can't guarantee this with the takeaway example we just looked at; we know we get that choice and control when we take the cook at home in our illustration, so the option with the most control is the *kitchen as a service* or *IaaS* model.

But maybe the pizza base is not of concern; you don't want the trouble of making your pizza, as that's time, effort, and skill required even to know how to make a

pizza base. So, you will let somebody else provide that for you, which will probably be better, faster, and cheaper than you could do, so you can focus on just choosing your toppings as this is where the value and fun bit is. Pizza bases are boring, but selecting the toppings is where it gets exciting, and you want to focus on the ingredients. This is where the *ingredients as a service* or *PaaS* model now has the most appeal.

The summary of all this is that each model will have its place that best meets your needs.

In the following section, we will introduce the concept of *serverless* computing to understand its positioning with the three service models of IaaS, PaaS, and SaaS that we looked at in this section.

What is serverless computing?

As this book's context is that of the knowledge and skills required to pass a Microsoft certification exam successfully, we should start with understanding Microsoft's definition of serverless:

"Serverless computing enables developers to build applications faster by eliminating the need for them to manage infrastructure. With serverless applications, the cloud service provider automatically provisions, scales, and manages the infrastructure required to run the code."

The term *serverless* itself is a bit of a misnomer, as in reality, there are servers involved, much like wireless does have wires involved at a certain point in the solution; it's more the fact that the servers do exist, but you don't need to know or care that they exist for you to have your desired outcome met. We come back to the topic of abstraction again; the same layers still exist, and there are still servers that execute the code passed down from the runtime layer – it's just that this layer is now further abstracted from you than it was in the IaaS and PaaS models.

The benefit of this further abstraction is that there are even fewer components to create and manage and allow development teams to focus on writing their core code without considering what's running their code; the provider takes care of automatically provisioning these resources to run the code. This all means faster, more productive development teams, less operational overheads for DevOps teams, more significant innovation, and quicker time to value and return on investment in development resources:

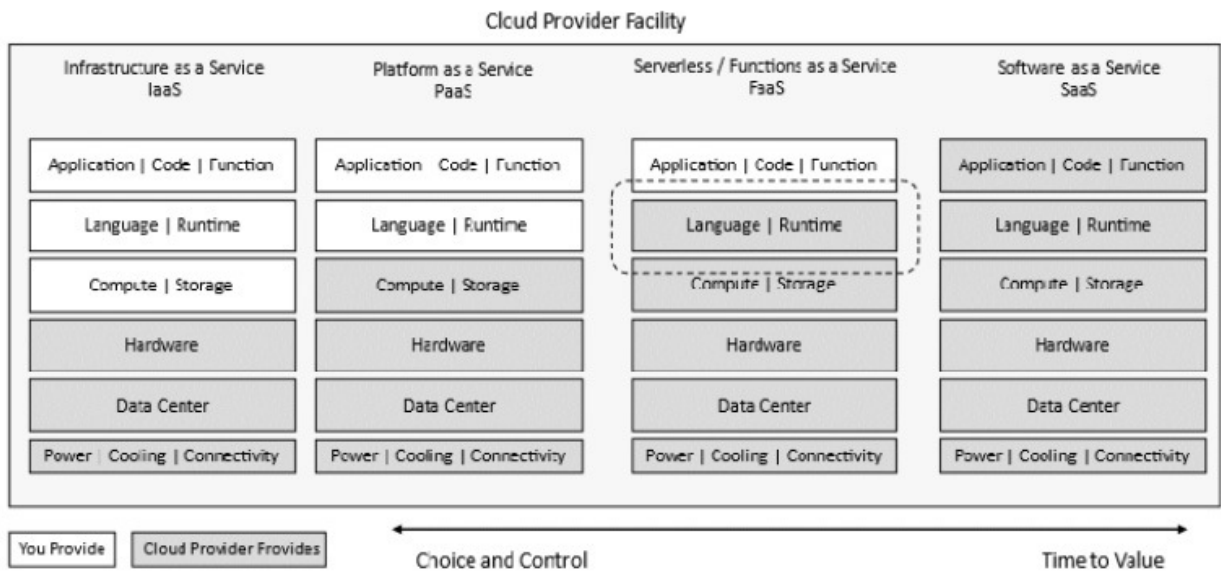


Figure 1.9 – Cloud computing service models – compare and contrast

In the following illustration, we again liken this to consuming pizza:

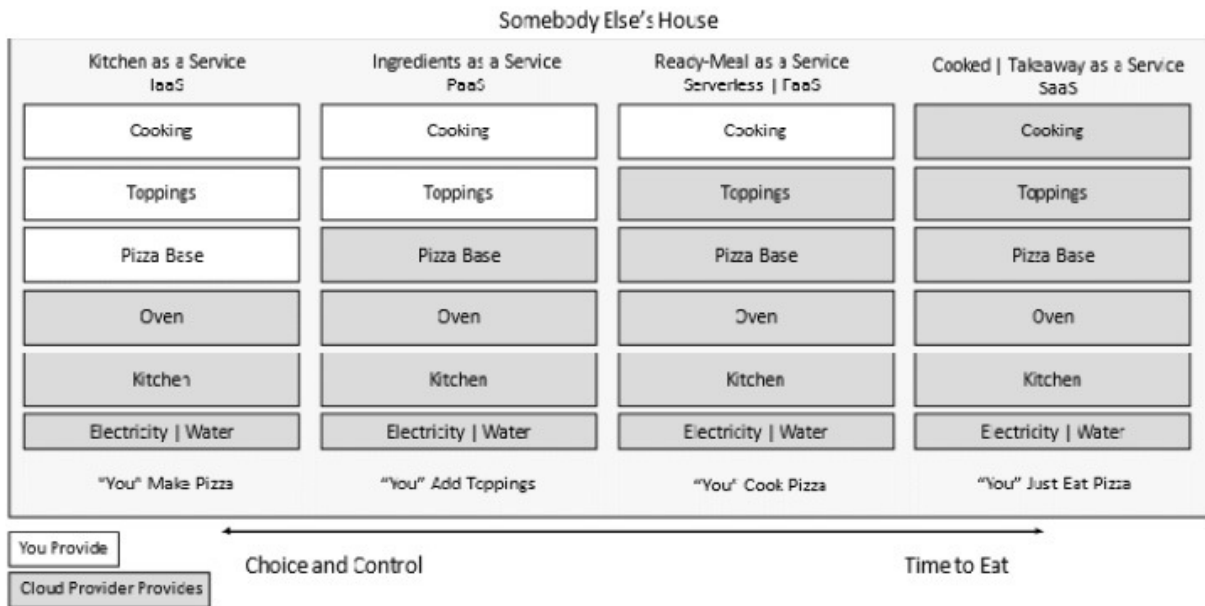


Figure 1.10 – Cloud computing service models – layer providers

In the preceding illustration, we have included the pizza analogy once again, this time to expand to include the *serverless* model. Again, the comparison here in this illustration to the cloud computing service model is to what level of input and control you want to fulfill that requirement to eat some pizza.

Some of the caveats of serverless are as follows:

- Event-based workloads are the best use case.
- Long-running tasks are not well suited.
- The execution environment cannot be customized.
- The cloud provider supports specific languages and runtimes.

Comparing the cloud computing service models

Each service model has its characteristics. The most appropriate model is defined by how much you want (or need/mandate) to control, secure, and manage your resources, for example, your apps, code, data, networks, security, and so on. From the last section, we can now define what the service models are; this section looks at the characteristics of each model in more detail to help you understand when you may choose one service model over the other.

Characteristics of IaaS

In a nutshell, IaaS is a model where you can host your virtual machines and infrastructure services on hardware provided for you and shared with other tenants.

The cloud provider is responsible for providing all layers up to and including the hardware; you are responsible for providing all layers preceding (refer to *Figure 1.10*).

The following are the characteristics of IaaS:

- You create the virtual machine (install an OS and software), storage, and computing resources as you would in a traditional on-premises computing model; this can be likened to a virtual data center. There is the ability to provide fault tolerance, redundancy through availability solutions in case of failure within an Azure data center, zone, or region.
- You have control to increase resources such as the processor, memory, and storage of a virtual machine by using self-service without requiring to redeploy or create a new virtual machine of the required spec.
- Based on the OpEx model, meaning you only pay for resources you consume on a *pay-as-you-go* basis; you only pay for a virtual machine while it's running, therefore your month-to-month running charges may be different across virtual machines if you run them for a different number of hours in the month. You will not be charged while it is in the stopped deallocated state; storage costs will still be charged if you wish to persist the data on disks.
- It provides the greatest control and flexibility to deploy, configure, manage, and support resources as you require and requires the most management and administrative and operations overhead.
- You have direct access and complete control of the virtual machine, the operating system, and any roles/services such as the web server, application server, or SQL server that may be

required to be installed/running on the virtual machines, as well as complete control over decisions on networking, security, and protection.

The following are examples of Microsoft IaaS resources:

- Azure Virtual Machines
- Azure Storage
- Azure Networking

Characteristics of PaaS

In a nutshell, PaaS is a model where you host your application, code, data, and business logic on compute, and storage resources are provided for you and shared with other tenants, but secured and isolated from other tenants.

The cloud provider is responsible for providing all layers up to and including the compute; you are responsible for providing all layers above (refer to *Figure 1.10*).

The following are the characteristics of PaaS:

- It provides a *ready-to-use* environment and platform for faster deployment of hosting web applications, code, business logic, data, and so on. Using pre-deployed resources, development frameworks, languages, and runtimes provided as a service, there is a quicker time to value and consumption of the service.
- It provides on-demand autoscaling of the platform used by a hosted application and services.
- You have control to increase resources by changing the pricing tier of the service by using self-service; you must still select underlying compute resources sizing to host your app, code, and so on.
- You would have no direct access or control of the virtual machine or any applications, services, or roles that may be installed; you do not get to specify or control which versions are available.
- Because the service provider is responsible for the compute and storage resources layer, it gives you the least control and least flexibility. Still, it requires the least amount of management, administrative, and operations overhead.

The following are examples of Microsoft PaaS services:

- Azure App Service
- Azure SQL Database
- Cosmos DB

- Azure Files
- **Azure Active Directory Domain Services (AADDs)**

Characteristics of FaaS (serverless)

In a nutshell, FaaS is a model where you provide your code and business logic and the cloud provider hosts it in their language, runtime, and compute environment shared with other tenants.

The cloud provider is responsible for providing all layers up to and including the language, runtime, and compute. You are responsible for providing all layers above, that is, the business logic layer (refer to *Figure 1.10*).

The following are the characteristics of FaaS:

- Azure Functions is a serverless code engine. It has the use case of events that trigger code; that is, Functions code being executed is a response or action based on an event trigger.
- Azure Logic Apps is a serverless workflow engine. It has the use case of events that trigger workflows; that is, a Logic Apps workflow is a response or action based on an event trigger.
- You only have control over your application, code, and business logic layer; all other layers are provided as a service that you have no access or control over. Essentially, you take the layers supplied to you and use that to execute (run/launch) your code/workflow. Using the pizza analogy covered earlier, this is much like what could be described as a *take and bake* approach; ultimately, you are still responsible for cooking, but they have provided all the layers below, so it's ready to cook, as it were, much like a microwave meal, or taking the packaging off a pizza and putting it straight in the oven.
- It provides the least control and flexibility but abstracts all the layers below, providing the least amount of deployment, configuration, and maintenance, so you can focus on the application, code, and business logic layers, where the value is, without needing to concern yourself with the layers below.

The following are examples of Microsoft Serverless resources:

- Azure Functions
- Azure Logic Apps

Characteristics of SaaS

In a nutshell, SaaS is a model where you consume an application provided for you and shared with other tenants.

The cloud provider is responsible for providing all the layers up to and including the applications; you are not responsible for any layers above other than the configuration and consumption of the app (refer to *Figure 1.10*).

The following are the characteristics of SaaS:

- The cloud provider installs the application/solution and is responsible for its updates, scalability, availability, and security.
- It provides the greatest time to value as there is no development time or resources required to create an application; it can be directly configured as needed and used instantly.
- In our pizza analogy, all the layers are taken care of for us, so all we need to do is just eat; this is the classic *takeaway* approach.
- It provides the minimum amount of control and input on the lower layers.

The following are examples of Microsoft SaaS solutions:

- Microsoft Teams
- Microsoft Exchange Online
- Microsoft SharePoint Online
- Microsoft OneDrive
- Microsoft Dynamics 365

The following are examples of other vendors' SaaS solutions:

- Zoom
- Salesforce
- Dropbox
- Google Mail/Google Docs

In this section, we covered the service models of IaaS, PaaS, and SaaS and introduced the concept of serverless computing as an extension of PaaS. We compared and contrasted each service model and outlined the characteristics unique to a particular model and those common across the models.

Summary

This chapter included complete coverage of the AZ-900 Azure Fundamentals exam skills area *Describe Cloud Concepts*.

We described what cloud computing is, where the platform environments and architectures have evolved from, and their direction of travel. We then looked at the Shared Responsibility model, which is critical to understand the security model when adopting cloud computing. We concluded by outlining the delivery and service models for cloud computing, comparing each, outlining the characteristics, and including some simple examples; this helps us to understand the use case for each model and which model may be most appropriate in any given scenario.

Further knowledge beyond the required exam content was provided to prepare for a real-world, day-to-day Azure-focused role.

In the next chapter, we will look at the benefits and value of cloud computing and its positioning as a digital transformation enabler.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure Fundamentals: <https://docs.microsoft.com/en-us/learn/certifications/exams/az-900>
- Exam AZ-900: skills outline: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- *Describe core Azure concepts*: <https://docs.microsoft.com/en-us/learn/paths/az-900-describe-cloud-concepts>
- *Describe the different types of cloud computing*: <https://docs.microsoft.com/en-us/learn/modules/fundamental-azure-concepts/types-of-cloud-computing>
- *Describe the different categories of cloud services*: <https://docs.microsoft.com/en-us/learn/modules/fundamental-azure-concepts/categories-of-cloud-services>
- Microsoft cloud computing definition: <https://azure.microsoft.com/en-in/overview/what-is-cloud-computing>
- NIST cloud computing definition: <https://www.nist.gov/news-events/news/2011/10/final-version-nist-cloud-computing-definition-published>

Skills check

Challenge yourself with what you have learned in this chapter (*Note down the answers that you believe are correct; go back and review the content of this chapter for any you are unsure of*):

1. List three public cloud computing platforms and their providers.
2. List three private cloud computing platforms and their providers.
3. Explain a hybrid cloud approach.
4. Explain the definition of cloud computing and Microsoft Azure.
5. What cost expenditure model is used for public, private, and hybrid cloud models?
6. Does adopting a public cloud platform mean you no longer need an on-premises data center?
7. List these architectures in order of evolution: containers, serverless, bare metal, virtual machines.
8. Explain how containers compare to virtual machines.
9. Explain how PaaS compares to serverless (FaaS).
10. What are the three levels to be considered for the Shared Responsibility Model?
11. List three cloud computing delivery models.
12. List a minimum of three characteristics of public cloud computing resources.
13. List a minimum of three characteristics of private cloud computing resources.
14. List a minimum of three characteristics of the hybrid cloud computing approach.
15. List four cloud computing service models.
16. Explain serverless computing.
17. List a minimum of two characteristics of IaaS.
18. List a minimum of two examples of IaaS resources.
19. List a minimum of three characteristics of PaaS.
20. List a minimum of two examples of PaaS resources.
21. List a minimum of three characteristics of serverless (FaaS).
22. List two examples of serverless/FaaS resources.

23. List a minimum of three characteristics of SaaS.
24. List a minimum of two examples of SaaS resources.
25. List a minimum of two non-Microsoft SaaS solutions.

Chapter 2: Benefits of Cloud Computing

In [Chapter 1](#), *Introduction to Cloud Computing*, you gained the skills needed to define cloud computing, describe the delivery and service models, and compare characteristics and use case scenarios.

This chapter will outline the benefits and value of cloud computing and its positioning as a digital transformation enabler, and we'll look at the audience of cloud computing and the cloud mindset that should be adopted compared to the traditional computing mindset. We will also cover the computing concept of a hierarchy of needs and look at cloud computing's operations model. The chapter closes with the economics of cloud computing and the foundational change of moving from a CapEx to an OpEx cost expenditure model.

The objectives for this chapter continue the coverage of the AZ-900 Azure Fundamentals exam skills area, *Describe Cloud Concepts*.

By the end of this chapter, you will be able to do the following:

- Identify the benefits of cloud computing and Microsoft Azure.
- Describe scalability, elasticity, agility, high availability, and disaster recovery.
- Describe the consumption-based model.
- Identify the differences between **capital expenditure (CapEx)** and **operational expenditure (OpEx)**.

In addition, this chapter's goal is also to take your knowledge beyond the exam content so you are prepared for a real-world, day-to-day Azure-focused role.

Why cloud computing?

Cloud computing adoption is often driven more by the advantages and benefits of its business and operational model than its technology model or location factors.

Cloud computing doesn't have to be a binary decision between a fully public cloud and no public cloud. The benefit of the cloud computing model is that it gives you a choice in how computing resources can be provided and consumed by a business to suit its operating model in the most appropriate way.

Cloud computing-based resources can be in the service provider's facilities or the customer facilities in the case of the Azure Stack portfolio (Edge, HCI, Hub), being a private cloud. Therefore, you can have on-premises cloud computing resources; it doesn't always have to mean resources in a provider's facility that are shared with others and accessible over a network, but that is the most typical deployment model, that is, the public cloud.

The core benefits of the cloud computing model over the traditional computing model are outlined in the following figure:

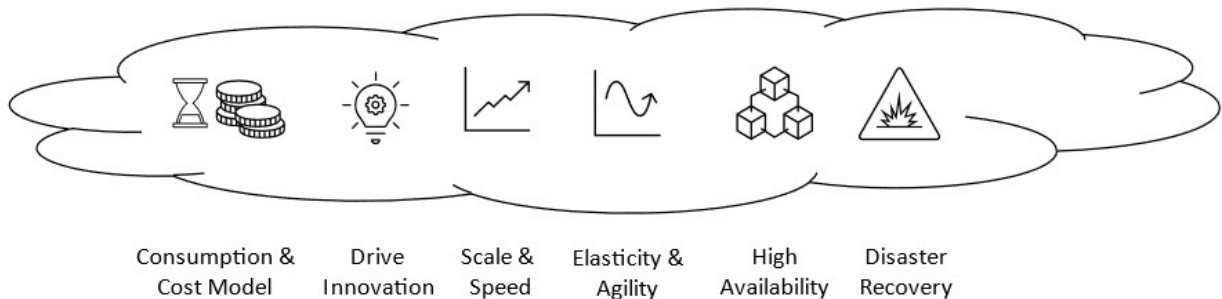


Figure 2.1 – Benefits of cloud computing

Other operational advantages include there being no need to replace failed hardware or address physical security for public cloud computing; the cloud provider manages the underlying hardware that hosts the resources. This means no CapEx to purchase hardware, although CapEx is still applicable in private cloud scenarios where hardware and facilities may be under your control; they may be at

a third-party hosting provider's facilities that provide the hosting for your private cloud.

However, there is a requirement in public cloud computing environments to provide security and protection for the customer systems and data, ensure it is redundant and available, protect the identities that access the systems and data, govern access permissions, and manage updates.

This section outlined the benefits of the cloud computing model. In the next section, we will look at cloud computing as an enabler and methodology rather than as tangible technology assets.

Cloud computing as a digital transformation enabler

Digital transformation is disruptive, a growth catalyst, and foundational in changing how an organization will deliver *time to value*.

The following figure outlines the value that a digital transformation enabler can realize for an organization:

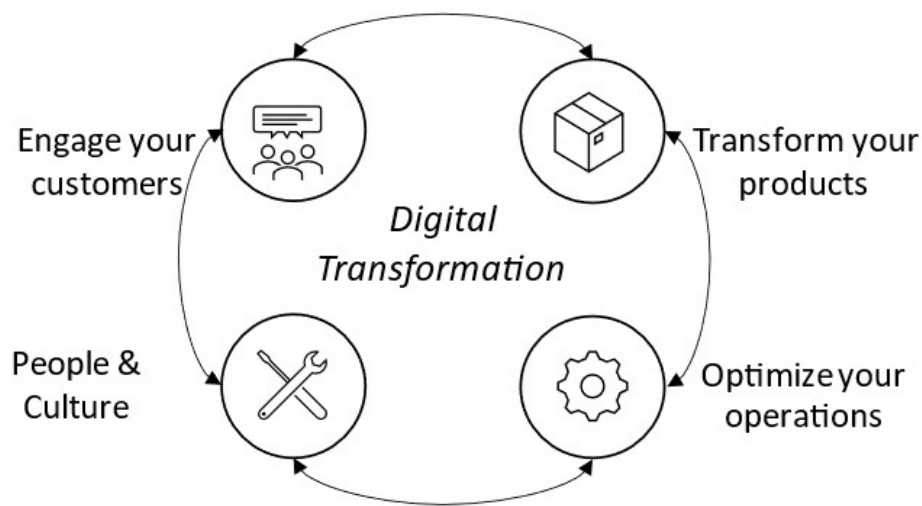


Figure 2.2 – Cloud computing as a digital transformation enabler

Cloud computing can be seen as a vital part of any *digital transformation* journey. Still, the reality is that it's less about the *technology model* and more about the *business model*, the *people*, and the *process*. It's a fact: people don't like change and are generally only forced to move direction when they have to; as in the analogy of placing a frog in a pot of boiling water, there must be a trigger to induce the action. The following section looks at those digital transformation triggers.

Digital transformation triggers

The reason to adopt any disruptive technology, especially one that can have a business impact, needs to start with a trigger relevant to the stakeholder's pain points or objectives and any business operations or technical operations drivers. This move often begins with a *business directive-led discussion* rather than a *technology-led* one; a business leader will always ask a technology leader *what the benefits are*, not *what the features are*.

Once triggers have been identified, an approach can be planned; you can think of the *triggers* as the *why* and the *approach* as the *how*.

The following illustration outlines some of the business-led as well as technology-led triggers for digital transformation initiatives:

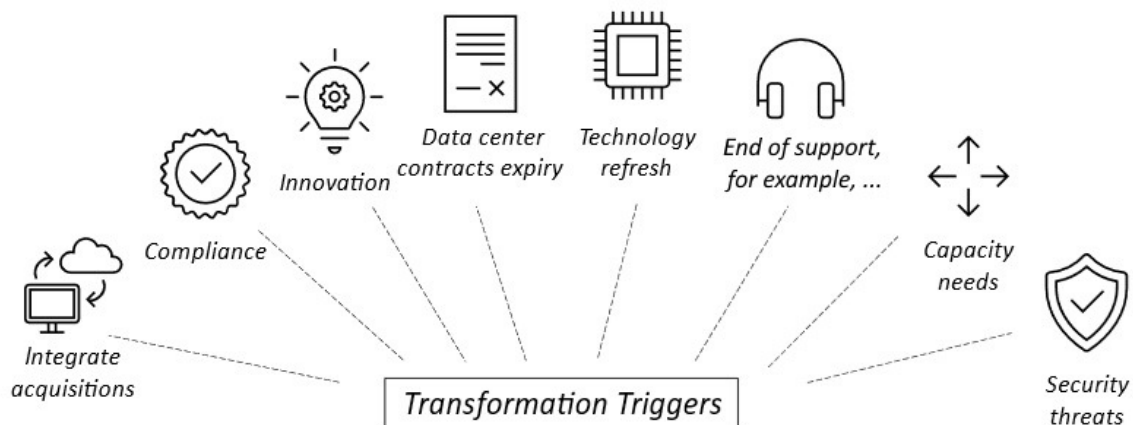


Figure 2.3 – Digital transformation triggers

In this section, we looked at *digital transformation triggers*. The following section looks at the migration approach.

Migration approach

The next step in the transformation journey is to understand the processes, methods, and services that could be used to execute the transition to cloud services; this is outlined in the following list:

1. **Phase 1 | Assess** – This phase is an exercise in understanding the business operations and technical operations of an organization. It starts with discovery, which provides an inventory of apps, data, infrastructure, and, critically, any dependencies. This will help evaluate the best strategy to take for each area of the captured inventory.

Example services: Azure Migrate Assessments.

2. **Phase 2 | Move** – This phase is the physical exercise of moving inventory items to their cloud services target, for instance, moving to SaaS, IaaS, PaaS, or serverless cloud computing services.

Example services: Azure Migrate Migrations, Azure Blueprints.

3. **Phase 3 | Optimize** – This phase will occur after a period of *bedding-in*, typically a period of months; this may include right-sizing activities to better suit the workload for costs and performance benefits.

Example services: Azure Advisor, Azure Cost Management.

4. **Phase 4 | Secure and Manage** – This is an ongoing operations phase and should include governance and control, security, and the protection of the network, app, data, and identities.

Example services: Azure Monitor, Azure Security Center, Azure Defender, Microsoft Cloud App Security, Azure Policy, Azure Sentinel, Azure Backup, Azure Site Recovery, Azure Firewall, Azure App Gateway, and Azure Front Door.

The Azure services listed in the preceding list will be covered in more detail later in this book.

This section looked at the migration approach. The following section looks at the audience of cloud computing.

Cloud computing's target audience

Cloud computing can be different things to different people depending on their perspective and their role or function. A business leader will have one idea of what cloud computing needs to deliver, which will be very different from the technology leader and very different from a developer, a database admin, or a data scientist. They all have other wants and needs, and so cloud computing as a model almost has to be all things to all people.

Azure provides the following high-level categorization of services to a business:

- Hosted infrastructure platform
- Application development platform
- Data and analytics platform
- Monitoring and management platform
- Hybrid and multi-cloud control plane platform
- Security operations platform
- Identity and compliance platform

With regard to Azure, as Microsoft's cloud computing platform, these different functions are very well catered for, making it relevant from all perspectives.

Azure is the only hybrid-consistent platform, combining Azure as the *public cloud platform* with Azure Stack as the *private cloud platform*, while operating with other cloud platform providers and on-premises resources through **Azure Arc**.

This section looked at the audience of cloud computing and the categorization of services available to a business. In the next section, we look at the mindset of the cloud computing audience compared to traditional computing's audience; this is a shift in people, process, and culture, which is much harder to achieve than the adoption of technology.

Cloud computing mindset

The biggest challenge to cloud adoption is not technology but changing the mindset and culture within an organization. We have seen the same thing happening with DevOps over the years. The realization over time has been that DevOps is not a technology or a job title but a cultural shift in a people and process model. Terms such as CapEx and OpEx referenced in the following section will be explained later in this chapter.

What is a traditional computing model mindset?

The following outlines what we see as a traditional computing model mindset:

- **Capital Expenditure (CapEx)** cost expenditure model.
- (Over-) Provision for peak demand and build in 5-year growth from day one.
- Fixed in size and scaling approach.
- Order 3 months before you need it; procurement lead times and project deployment times.
- A monolithic, tightly coupled infrastructure stack approach.
- Always-on 24/7 operation.

In the following section, we will compare and contrast the traditional computing model mindset with the cloud computing mindset.

What is a cloud computing mindset?

The following outlines what we see as a cloud computing model mindset and contrasts with a traditional computing mindset:

- **Operational Expenditure (OpEx)** cost expenditure model.
- Usage versus provisioning; just-in-time, demand-driven provisioning.
- Cloud computing is designed to be elastic, scale in and out, and burst to meet demand.
- Consume and pay for what you use for as long as you need it when you need it; shut down or pause when you don't.
- A microservices, loosely coupled, business logic-centric approach.
- A cloud-agnostic thought pattern.

This section looked at the cloud computing mindset. The following section looks at the concept of the cloud computing hierarchy of needs.

Cloud computing hierarchy of needs

The IT services delivery model can draw parallels to *Maslow's hierarchy of needs*; this comes from a psychology theory, often represented in a hierarchical pyramid. In this model, the lower-level needs add no value or benefit, but each lower-level need must be met before the next-level needs can be met, leading to dependencies on the lower levels. The challenge is a very delicate balance in that any change in the lower levels affects the chances of success or failure of the required *outcome*; in this theory, the *outcome* to be achieved is *self-actualization*.

This idea can be applied to the *IT services delivery model*. In this model, the typical technology stack provides the *outcome* that needs to be met, such as an *app*, *function*, or *code*...:

- ...this must exist on a lower layer of *software*...
- ...which needs to live on a lower layer of *compute*...
- ...which needs to exist on a lower layer of *hardware*...
- ...which needs to exist in a *physical facility*...
- ...which needs power, cooling, *physical* security, facilities management, staffing, and so on.

As with Maslow's theory, the top-level *needs* depend on the lower levels being fulfilled.

With the traditional computing approach, a large amount of the budget is not used for innovating but just to *keep the lights on*. Resources and budget are trapped at the bottom, being used to procure hardware and invest in data center facilities, not being utilized or benefiting from being invested in the higher value-driver layers offering real value to the business. It is a CapEx constrained model, and technology is seen as a *cost center* to a business and not an *innovation center* for the business:

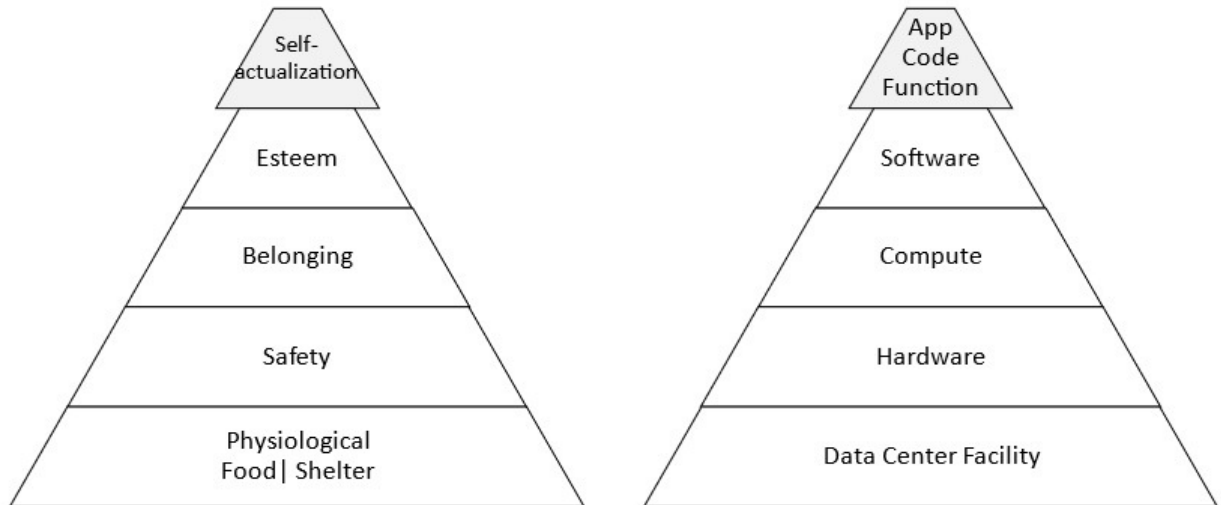


Figure 2.4 – Maslow's hierarchy versus the cloud computing hierarchy of needs

The cloud computing model aims to re-balance resource utilization and budget, allowing the cloud services provider to spend *their* budget and use *their* resources on the hardware and lower layers. This means a business can ensure its time is more effectively spent at the top, *innovating*, and not at the bottom, *maintaining*. The goal is to move to a flexible and agile OpEx model, consuming only what is required from each layer down to deliver the business *need* and *value-driven outcome*.

This section looked at the cloud computing hierarchy of needs. The following section looks at the computing operating model, one of the key differentiators between traditional computing and cloud computing.

Cloud computing operations model

Elastic, scalable, agile, fault-tolerant, highly available, and disaster recovery are all terminology associated with cloud computing, adding value, and benefiting an organization's operational model.

These inherent and defining characteristics allow a workload deployed into a cloud computing environment to become highly available and scale in and out (both vertically and horizontally), which maps closely to demand. This ability to be elastic in nature allows the agility to provide a highly effective operations and economics model to flex with the changing demands of a business.

By *optimizing running hours and right-sizing resources* in line with demand and changing requirements, the switch to an OpEx system of paying as you use things (also known as a *consumption-based system*) allows controlled and monitored spending of computing resources without the overcommit of a traditional CapEx computing cost model of deploying hardware in on-premises facilities.

The following figure outlines the computing resources demand model and shows the implications of actual demand against implemented resources based on predicted demand:

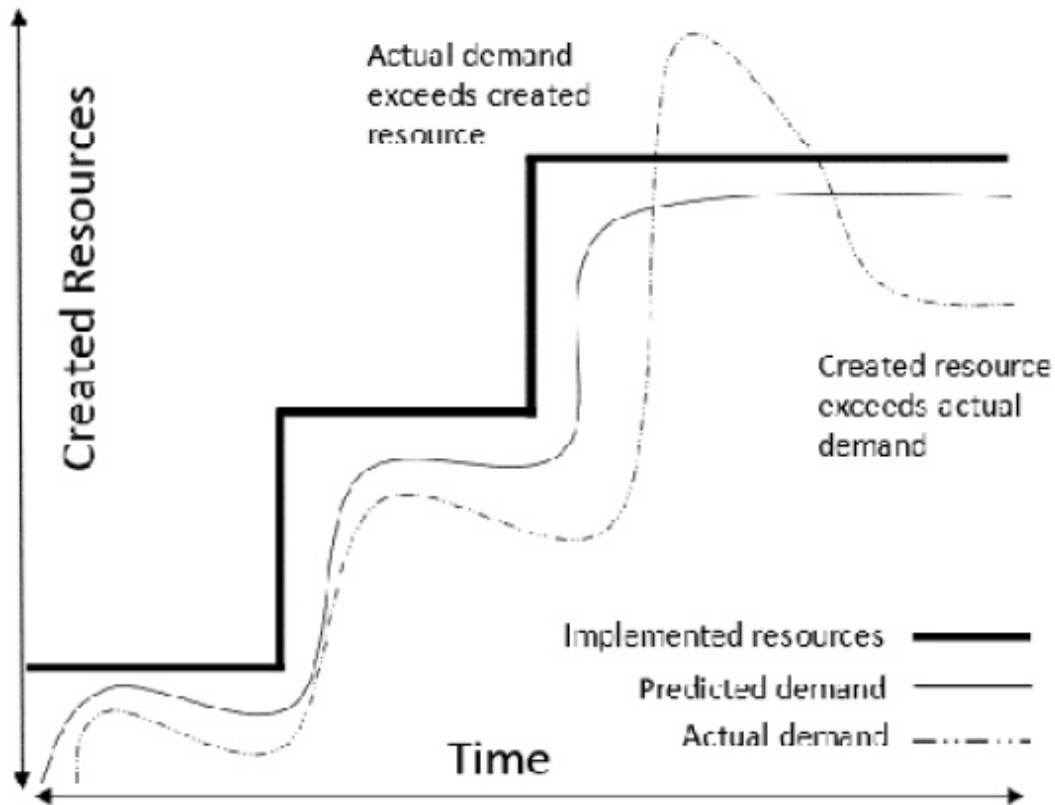


Figure 2.5 – Cloud computing resource demand model

In this illustration, the *traditional computing mindset* from the last section can be seen. This traditional computing mindset means over-provisioning to provide resources to meet predicted demand; many resources are underutilized. When demand exceeds the predicted demand, there are no resources available; there is no burst capacity or scale to meet the demand. To compound things further, this demand has dropped off by the time these extra resources have been implemented and they are no longer needed.

With the *cloud computing mindset*, resource utilization can be tracked and then *right-sized* to demand, meaning there is never a case where it is required to *over-provision* and pay for more resources than are needed.

This section introduced the cloud computing operations model. The following section will look closer at the characteristics of cloud computing that deliver the benefits and value over the traditional computing model.

Operational benefits of cloud computing

This section will look at the operational benefits that cloud computing can add to an organization over those provided in a traditional computing model.

Cloud computing platforms primarily provide the following operational benefits over traditional computing models:

- Scalability
- Elasticity
- Agility
- High availability (and geo-distribution)
- Disaster recovery
- Cost-efficient

The listed operational benefits may be an inherent built-in platform function that provides features as part of the service, as is typically the case with *PaaS* (or *FaaS* and *SaaS*). These operational benefits just need to be enabled in some cases, if not an automatically included part of the services.

The listed operational benefits could also be something that needs to be designed into part of the solution as an individual set of resources that need to be implemented to enable these characteristics.

For example, *IaaS virtual machines* will not provide scale, elasticity, high availability, and disaster recovery without this being designed into the solution and then implementing resources to provide the functionality to meet each of these characteristics.

For *PaaS* (*FaaS* and *SaaS*), the takeaway is that the cloud platform provider will generally provide these functions and characteristics; you may layer on additional functionality as your needs dictate.

Of course, not everything is perfect with the cloud computing model; the following are some challenges that can be overcome but must be considered and provided

for:

- Network dependency: reliability, stability, quality, and performance
- **Confidentiality, Integrity, Availability (CIA)** of users, apps, and data
- Access control and operational governance
- Cost control

This section covered the operational benefits and challenges to be considered. In the following section, we will look at the benefits of cloud computing in more detail.

What is scalability?

Scalability refers to how to increase resources based on demand, usually in an automated way triggered upon a metric such as a time or resource threshold being reached:

- **Scaling up** (*vertical scaling*) means capacity is increased within the resource, such as increasing processor or memory by resizing a virtual machine.
- **Scaling out** (*horizontal scaling*) means additional resource instances, such as adding other virtual machines or compute node/scale units.

What is elasticity?

Elasticity refers to the ability to shape the resources needed automatically, burst and scale to meet any peak in demand, and return to a normal operating baseline.

What is agility?

Agility means deploying and configuring resources effectively and efficiently in a short space of time to meet any change in requirements or operational needs.

What is high availability (and geo-distribution)?

High availability and **geo-distribution** mean deploying resources to operate within the required or mandated **Service Level Agreement (SLA)** for those resources. An SLA is a guaranteed measure of *uptime*, meaning the amount of time the services are online, available, and operational.

Microsoft defines an SLA as follows: *The Service Level Agreement (SLA) describes Microsoft's commitments to uptime and connectivity.*

Microsoft provides different SLAs for all services resources or services individually; the service availability depends on the number of nines (as in the three nines in "99.9%") of the SLA.

Availability is the *percentage of time* a resource is *available to service requests*. *Service availability* is expressed as the *uptime percentage over time*, for example, 99.9%. Microsoft SLAs are expressed on a monthly basis, so this would have an allowed service downtime of 43.2 minutes *per month*. Increasing availability often results in increases in costs due to the complexity of the solutions required to deliver the level of availability.

Availability depends on *resilient systems*, meaning the ability of a system to recover from failures and continue to function.

Failover is also another critical factor in availability; this is where one system takes over from another when a resource fails and becomes unavailable and is part of an availability and disaster recovery strategy.

What is disaster recovery?

Disaster recovery refers to a set of initiatives designed to ensure that critical business systems will be protected, despite serious incidents, and recovered to an operational state within a defined period; this is achieved by failing over to systems that have been replicated in another region.

A disaster recovery strategy will be determined by the required **Recovery Time Objective (RTO)** and **Recovery Point Objective (RPO)**; replication technologies allow for much shorter RTOs and RPOs than can be achieved with backups:

- *RTO* refers to the maximum duration of acceptable *downtime* for the system.
- *RPO* refers to how much *data loss* is acceptable to a system:

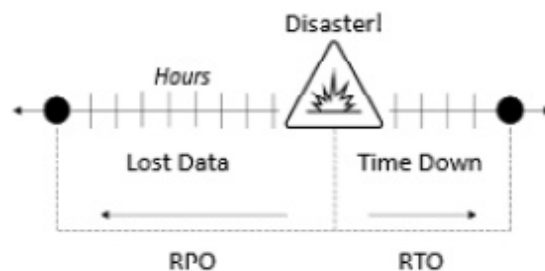


Figure 2.6 – RTO and RPO

This section covered *scalability, elasticity, agility, high availability (and geo-distribution), and disaster recovery*. In the following section, we will compare disaster recovery to the concepts of *high availability* and *backup*.

Comparing disaster recovery, high availability, and backup

High availability and disaster recovery could be classified as *systems protection*, whereas backup could be classified as *data protection*:

- **High availability** – When systems fail and are not available, you can run a second instance in the same Azure region.
- **Disaster recovery** – When systems fail and are not available, you can run a second instance in another Azure region.
- **Backup** – When data is corrupted, deleted, lost, or irretrievable (ransomware), you can restore the instance from another copy of the system.

The following illustration outlines the three preceding points of high availability, disaster recovery, and backup:

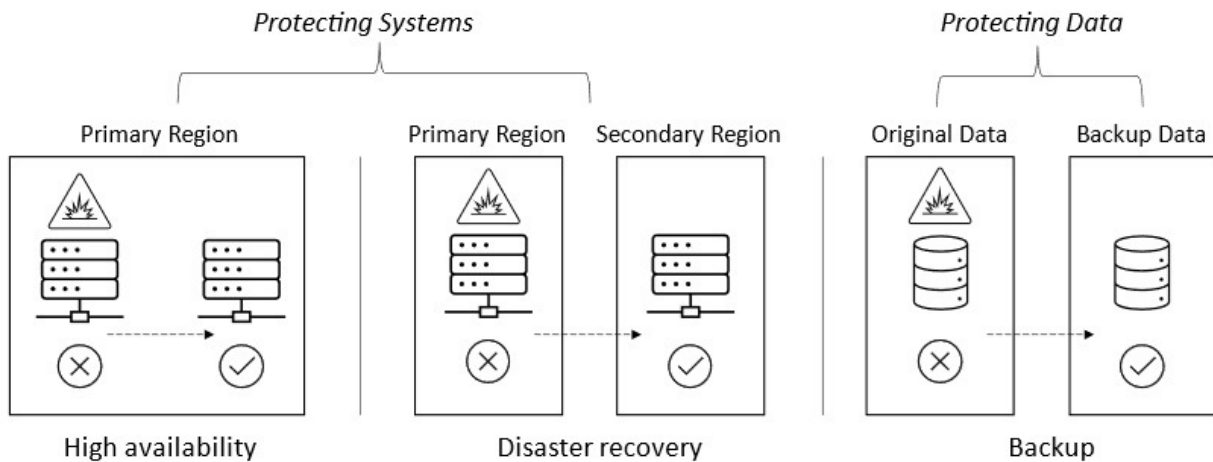


Figure 2.7 – Comparing backup, high availability, and disaster recovery

High availability, disaster recovery, and backup should not be an either-or decision in a strategy for *business continuity* and should include all three; they serve different purposes.

Challenges of implementing business continuity

Cost, complexity, and compliance are the biggest challenges; the result is that systems are often not covered by disaster recovery or protected through backups, which challenges your ability to comply with any regulatory or internal mandatory policy.

While you can be familiar with the traditional causes of a disaster or business disruption, the one that had never appeared until 2020 was a threat to business operations from a *global pandemic*, with mitigation and planning for a pandemic having never been included in any disaster recovery or business continuity strategy. While not a disaster or outage, a pandemic certainly causes a significant business disruption that almost nobody can foresee. It's reasonable to say that those who had already adopted some form of cloud services and a remote working strategy before the COVID-19 pandemic were probably better prepared than others.

The following illustration shows that when you adopt a cloud computing model, your costs are reduced, your complexity lowers, and your compliance levels increase:



Figure 2.8 – Challenges to implementing business continuity

Adopting a cloud strategy utilizing Microsoft Azure can address many of these challenges. The key benefit and driver is often cost. There is no need to maintain a secondary site and purchase the resources required for a secondary site. With Microsoft Azure as the secondary site, only what is used is paid for and it's paid for on an OpEx model instead of a CapEx model.

This section covered the operational benefits of cloud computing. The following section will cover the economics of cloud computing, the consumption model, and the cost expenditure model.

The economics of cloud computing

We'll look at the *consumption-based* model, one of the two economic characteristics of cloud computing compared to traditional computing economics; the second economic characteristic that cloud computing is based upon is the cost expenditure model of OpEx.

Consumption-based model?

In a nutshell, a consumption model means *paying only for what you use, for the time you use the resource*; this can be likened to leasing/renting something instead of outright purchasing and owning the asset. Some resources such as virtual machines can be stopped and started to reduce costs, so you only pay while running them; this is one of the key business benefits of the cloud computing cost model over a traditional computing cost model.

This section introduced the consumption-based model of cloud computing. The following section takes a closer look at the cost expenditure models.

Defining the expenditure models

It is essential to define some finance terms to understand OpEx and CapEx:

- **CapEx:** The upfront commitment of a large amount of money to purchase assets such as investment in data center facilities, network circuit implementation, physical hardware, or software, as a one-off payment that you then own for the lifespan of those assets. *You can liken this to the model of purchasing a vehicle or a mobile phone handset outright with a sum of money. You own it until it needs replacing, so you have to find another lump sum of money to reinvest to purchase another one outright in a few years.*
- **OpEx:** Essentially, a *pay as you go* model for consuming assets and resources. These are the running costs and are ongoing, which require a recurring payment when the asset or resource is required and in use to deliver service or functionality to a business. This could include staffing costs, data center facility management costs, power, and software that is not purchased outright but leased on a subscription basis. There is no upfront commitment of money to buy assets; you pay every month as you use the assets, only for the amount you use during a certain period. *You can liken this to the model of renting a vehicle or mobile phone handset and paying every month. You never own the asset; you use it for as long as you need, and you may swap it, upgrade, and downgrade as you wish within the terms:*

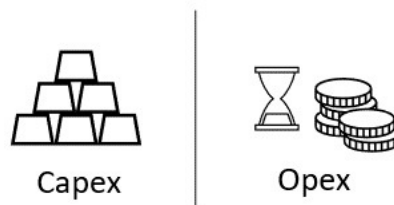


Figure 2.9 – Cost expenditure models

The value of cloud computing to a business is not so much the technology but the economic model it provides. The utilization of an *OpEx* model instead of the *CapEx* cost expenditure model associated with traditional computing is generally the value driver, with the technology being secondary.

This section outlined the cost expenditure models for a business that apply to cloud computing and traditional computing models. The following section looks at how those expenditure models are used and the benefit and value of cloud computing's OpEx model.

Applying cost expenditure models to cloud computing

In the traditional computing model (pre-cloud technologies), hardware resources and software were purchased up front in a one-off lump sum; these business assets would be seen as depreciating assets. This is the CapEx model.

To contrast this to the public cloud computing model, resources that are provided by the cloud platform are shared with others; this means there has been no CapEx to provision hardware, and so you can start using these resources on-demand. These are treated as day-to-day OpEx. The exception is reserved instances; in this scenario, you can commit to paying up front (or monthly) on a CapEx basis to reserve predicted resource usage. This may be more cost-effective over, say, a 3-5 year period than paying on the pay-as-you-go consumption model.

In the case of the private cloud computing model, this is cloud computing technologies hosted on dedicated, single-tenant hardware resources; typically, this is self-hosted on hardware in a facility (or colocation), but may also be hosted with a third-party hosting provider. In this model, there is an element of CapEx and OpEx, the most significant portion being CapEx for purchasing hardware and assets, with usage (and often licensing and software) as OpEx.

The hybrid cloud approach will give the greatest flexibility in the expenditure model, allowing you to decide which resources utilize which expenditure model to best fit the business's needs:

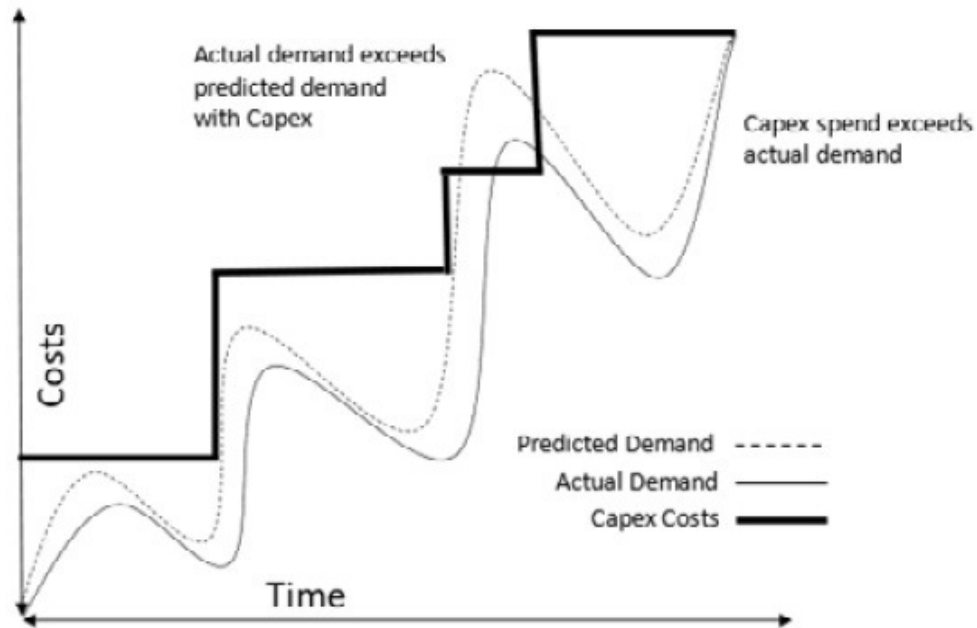


Figure 2.10 – Application of cost expenditure models

The consumption cost model of cloud computing means not needing to commit large amounts of capital expenditure on depreciating assets. Instead, by keeping that money within the business to use for day-to-day operations expenses and paying for what you need, when you need it, and only for as long as you need it, you can keep your costs much closer to actual demand, rather than over-committing to costs for a predicted demand. When resource demand exceeds predicted demand, the nature of the cloud computing model means you have resources that can be scaled in and out to meet that demand.

This section looked at how the cost expenditure models are applied and the benefit and value of the OpEx model of cloud computing. The following section covers a cloud computing migration use case scenario.

Thought exercise

This section allows you to reflect on all the information presented in this chapter and apply it in a customer requirements scenario.

Migration assessment discovery output

The following inventory was discovered from an organization assessment. *What approach should you take for each of the inventory items listed?:*

- Active Directory and DNS – Windows Server 2008 R2
- Email – Exchange Server 2010
- Intranet – SharePoint Foundation 2010
- User shares – Windows Server 2008 R2
- Company shares – Windows Server 2008 R2
- Line of Business Apps (App Servers) – Linux
- Line of Business Apps (Database Servers) – MySQL
- Remote Desktop Services – Windows Server 2008 R2
- Backup – Systems Center Data Protection Manager 2010
- Fax Server – Windows Server 2008 R2
- VPN – Firewall
- NAS – QNAP

Now that you have identified an approach to take for each of the inventory items listed, the following section will look at the solution proposal.

Migration solution proposal

The following represents the strategy that will be taken for each of the discovered inventory items:

- Active Directory and DNS – Windows Server 2008 R2

Proposed Strategy: *Rehost* as Azure IaaS VM; investigate future refactor to Azure AD Domain Services.

- Email – Exchange Server 2010

Proposed Strategy: *Replace* with Exchange Online.

- Intranet – SharePoint Foundation 2010

Proposed Strategy: *Replace* with SharePoint Online.

- User shares – Windows Server 2008 R2

Proposed Strategy: *Replace* with OneDrive for Business or Azure Files.

- Company shares – Windows Server 2008 R2

Proposed Strategy: *Rehost* as Azure IaaS VM; investigate future *refactor* to Azure Files.

- Line of Business Apps (App Servers) – Linux

Proposed Strategy: *Rehost* as Azure IaaS VM; investigate future *refactor* to PaaS as Azure Database for MySQL.

- Line of Business Apps (Database Servers) – MySQL

Proposed Strategy: *Rehost* as Azure IaaS VM; investigate future *refactor* to PaaS as Azure App Service.

- Remote Desktop Services – Windows Server 2008 R2

Proposed Strategy: *Refactor* as Microsoft **Windows Virtual Desktop (WVD)**.

- Backup – Systems Center Data Protection Manager 2010

Proposed Strategy: *Replace* with Azure Backup.

- Fax Server – Windows Server 2008 R2

Proposed Strategy: *Retire.*

- VPN Server – Firewall

Proposed Strategy: *Replace* with Azure VPN Gateway and Azure Firewall; alternatively, investigate third-party vendor **Network Virtual Appliance (NVA)**.

- NAS – QNAP

Proposed Strategy: *Refactor* as Azure Files or **Azure NetApp Files (ANF)**.

This section involved a thought exercise to look at the migration approach applied to an organization's inventory.

Summary

This chapter included complete coverage of the AZ-900 Azure Fundamentals exam skills area, *Describe Cloud Concepts*.

We described the *benefits* and *value* of cloud computing, its positioning as a *digital transformation enabler*, and the *audience* of cloud computing. We saw that cloud computing is often more of a *mindset*, and we contrasted it to a traditional computing mindset. We then looked into how computing has a hierarchy of needs, how this evolves with the adoption of cloud computing, and we continued to look at cloud computing as an *operations model*.

The final exam topics skills covered were the *economics* of the cloud and the fundamental change of moving from a *capital expenditure model* to an *operational expenditure model*, closing with the idea that cloud computing is as much about the *people, processes, and business stakeholders* as it is about the *technology*.

Further knowledge beyond exam content was provided to prepare for a real-world, day-to-day, Azure-focused role.

The chapter concluded with a thought exercise bringing together all the skills areas covered in this chapter.

The next chapter will outline the *core architectural components* from both a *physical* and *logical* perspective. From the *physical* component perspective, we will look at the data centers that host the cloud computing resources, the global networks connecting them and connecting users to their resources, the global regions that provide the cloud platform resources, and the availability of these resources. From the *logical* component perspective, we will look at all aspects of *resource management*.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure fundamentals: <https://docs.microsoft.com/en-us/learn/certifications/exams/az-900>
- Exam AZ-900: skills outline: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- Discuss Azure fundamental concepts: <https://docs.microsoft.com/en-us/learn/modules/fundamental-azure-concepts>
- Learn the business value of Microsoft Azure: <https://docs.microsoft.com/en-us/learn/paths/learn-business-value-of-azure>
- Describe high availability and disaster recovery strategies: <https://docs.microsoft.com/en-us/learn/modules/describe-high-availability-disaster-recovery-strategies>
- Microsoft **Total Cost of Ownership (TCO)** calculator: <https://azure.microsoft.com/en-us/pricing/tco/calculator/>
- Microsoft Service Level Agreements: <https://azure.microsoft.com/en-us/support/legal/sla/>

Skills check

Challenge yourself with what you have learned in this chapter:

1. List six benefits of cloud computing platforms.
2. List three core audiences of cloud computing platforms.
3. List a minimum of two other audiences of cloud computing platforms.
4. Explain what digital transformation is and its value.
5. List a minimum of two business-led triggers.
6. List a minimum of two technical led triggers.
7. List the six Rs of the digital transformation approach.
8. List a minimum of three of the mindsets for the traditional computing model.
9. List a minimum of three of the mindsets for the cloud computing model.
10. Explain the cloud computing hierarchy of needs.
11. Explain scalability, elasticity, agility, high availability, and geo-distribution.
12. Explain disaster recovery.
13. Explain the consumption model.
14. Explain the CapEx versus OpEx model.
15. Explain the benefits of the OpEx model.

Section 2: Core Azure Services

This section provides complete coverage of the knowledge and skills required for the *skills measured* of the *Describe core Azure services* section of the exam. We also cover knowledge and skills that go beyond the exam content, so you are prepared for a real-world, day-to-day Azure-focused role.

This part of the book comprises the following chapters:

- [Chapter 3](#), *Core Azure Architectural Components*
- [Chapter 4](#), *Core Azure Resources*

Chapter 3: Core Azure Architectural Components

In [Chapter 2, Benefits of Cloud Computing](#), you learned the skills to *identify the cloud computing benefits, describe the consumption-based model, and identify the differences between capital expenditure and operational expenditure.*

This chapter will outline the *core architectural components* from both a *physical* and a *logical* perspective.

From the *physical* component perspective, we'll look at the data centers that host the cloud computing resources, the global networks connecting them and connecting users to their resources, the global regions that provide the cloud platform resources, and the availability of these resources.

From the *logical* component perspective, we'll look at all aspects of *resource management*. Starting with **Azure subscriptions**, which act as both a mechanism and a boundary for billing and access management, we'll also cover **management groups**. Next, we'll cover **Azure Resource Manager (ARM)** and **resource groups**, which form the basis for access management and governance, the concepts of **Role-Based Access Control (RBAC)**, **Azure Policy**, how they differ, and the scenarios in which to use each.

This chapter aims to provide complete coverage of the AZ-900 Azure Fundamentals *Skills Measured* section *Describe Cloud Concepts*.

By the end of this chapter, you will have learned the following skills to be able to do the following:

- Describe the core architectural *physical* components such as *data centers, networks, regions, availability sets, and Availability Zones*.
- Describe the core architectural *logical* components such as *subscriptions, management groups, resource manager, and resource groups*.

To support your learning with some *practical skills*, we will also look at the *hands-on* creation of some of the resources covered in this chapter.

The following resources will be created.

- Exercise 1 – **Azure management groups**
- Exercise 2 – **Azure access assignment**
- Exercise 3 – **resource groups**
- Exercise 4 – **proximity placement groups**
- Exercise 5 – **availability sets**

In addition, this chapter's goal is also to take your knowledge beyond the exam content so you are prepared for a real-world, day-to-day Azure-focused role.

Azure global infrastructure

The key components that make up the Azure global infrastructure are the *physical data centers*, the *edge infrastructure*, and the *global network*, sometimes referred to as *premises* and *pipes*.

An *Azure physical data center* is a secure facility that hosts the physical compute, storage, and networking facilities that provide the Azure cloud computing platform resources.

Edge locations are secure facilities where traffic enters and leaves the Microsoft global network. These locations can provide *edge computing resources* to be closer to users for improved network latency, allowing fewer network hops through fewer providers, so the traffic can stay on the Microsoft backbone network longer, without transiting the internet if required. We will cover the concept of *cold potato routing* in [Chapter 4, Core Azure Resources](#), of this book.

The **Microsoft global network** is one of the largest private networks in the world. It connects data centers to regional gateways and edge locations where traffic can enter and leave the Microsoft **Wide Area Network (WAN)**.

Microsoft also provides **ExpressRoute**, a service that allows customers to create private network connections to Azure from specific peering locations, allowing traffic from a customer's **Multiprotocol Label Switching (MPLS)** WAN to enter the Microsoft network, bypassing the internet entirely and offering a low latency connection.

The following diagram outlines the Azure global infrastructure topology:

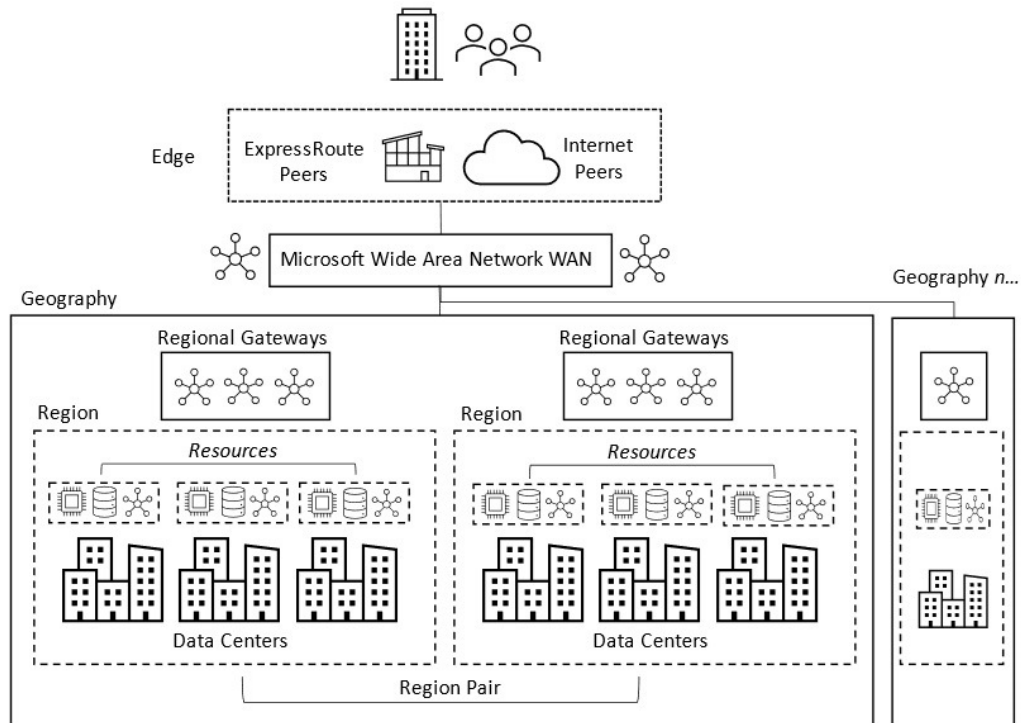


Figure 3.1 – Azure global infrastructure

The location for a resource will be the one that best meets the needs of the organization; it may be a technical driver, such as service capability and availability or latency in that region, or a business driver, such as compliance, data residency, or even cost.

This section provided an introduction to Microsoft's global infrastructure. The following section looks at Azure regions and geographies.

Azure regions and geographies

Microsoft data centers are grouped into collections across different sites to provide redundancy and high availability for the resources, such as compute and data hosted within the data centers.

These sets of grouped data centers are known as regions and are exposed as the deployment location when creating resources. There can be multiple regions for a continent, such as *Europe*, the *US*, and *Asia*, with multiple data centers in a region.

The following are some example regions within different geographies:

- *Europe geography:*
 - a) *North Europe region; located in Ireland*
 - b) *West Europe region; located in the Netherlands*
- *United Kingdom geography:*
 - a) *UK South region; located in London*
 - b) *UK West region; located in Cardiff*
- *US geography:*
 - a) *Central US region; located in Iowa*
 - b) *East US region; located in Virginia*
 - c) *East US 2 region; located in Virginia*
 - d) *East US 3 region; located in Georgia*
 - e) *North Central US region; located in Illinois*
 - f) *South Central US region; located in Texas*
 - g) *West Central US region; located in Wyoming*
 - h) *West US region; located in California*
 - i) *West US 2 region; located in Washington*
 - j) *West US 3 region; located in Arizona*

The complete list of geographies and regions can be found here:

<https://azure.microsoft.com/en-gb/global-infrastructure/geographies/>.

The data centers in a region are all connected by a low-latency network for availability, which provides the basis for *Availability Zones*, which we will cover in the next section.

Regions belong to a *geography* and are deployed in *pairs* for disaster recovery; they are located 300 miles or more apart where possible. This is so they can be far enough apart to be protected from local area disasters or service-impacting availability. Should a region have an outage, services will automatically fail over to the other region in the pair.

A *geography* can have multiple *regions* and defines a *data residency* and *compliance boundary*.

The following diagram outlines the Azure regions and geographies topology:

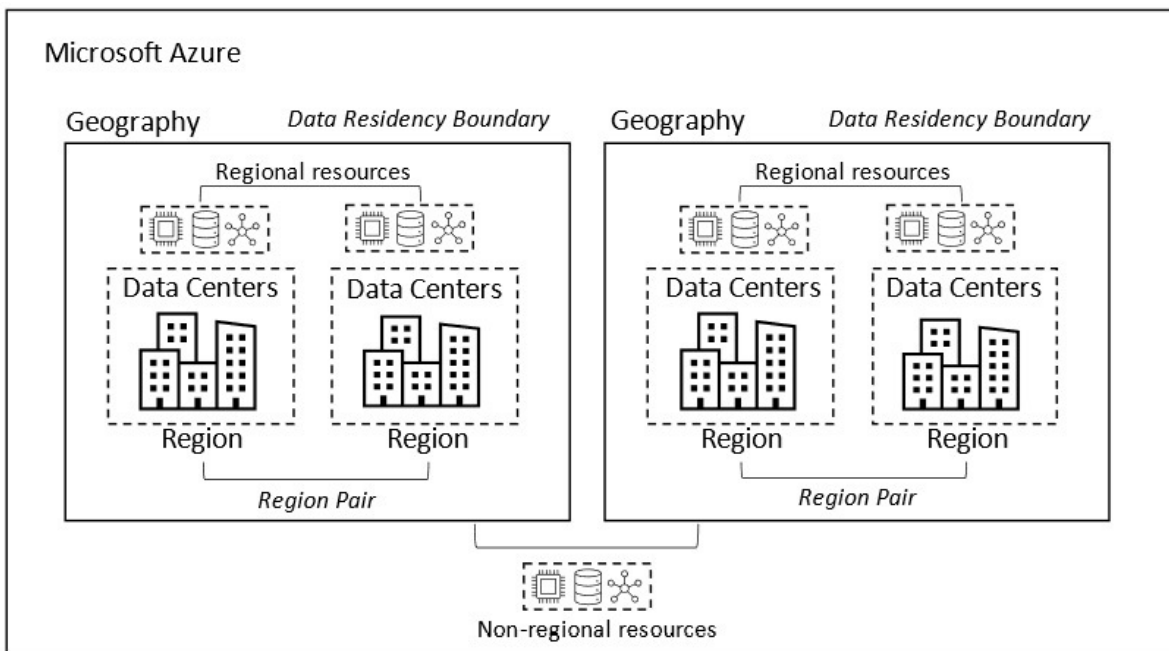


Figure 3.2 – Azure regions and geographies

Although Azure resources are physically located in data center facilities, they are not considered individual facilities to create resources within them. When creating an Azure resource such as a virtual machine, it is not possible to directly select a particular data center; only a region can be selected.

A resource, such as a virtual machine, will be created in any of the data centers in that region; these are *regional services*, as they are dependent on being collocated in that specific region's data centers.

Not all resources and services are available in all regions; new resources and services are only available in the US or a specific region and then rolled out to the rest of the world.

Some resources are *non-regional*, meaning they are not dependent on or bound to a particular region's data center and are a global service; however, that service's metadata can have a location that can be specified for compliance. Some examples of non-regional services are **Traffic Manager** and **Front Door**. In contrast, DNS and Azure **Active Directory (AD)** Domain Services are geo-based to ensure data is kept within that geography.

It should be considered when planning to create resources between regions that any data transfer leaving a region is billed; traffic within a region or ingress is free, but any region egress will be billed.

The following screenshot shows region selection only in the Azure portal when creating resources and not individual data centers:

Create a virtual machine ...

Instance details

Virtual machine name * ⓘ

Region * ⓘ

Availability options

Image * ⓘ

Azure Spot instance ⓘ

Size * ⓘ

Administrator account

Username * ⓘ

Password * ⓘ

Confirm password * ⓘ

Choose these locations for the broadest set of Azure products and long-term capacity growth

Recommended ⓘ

- (US) East US
- (US) East US 2
- (US) South Central US
- (US) West US 2
- (Asia Pacific) Australia East
- (Asia Pacific) Southeast Asia
- (Europe) North Europe
- (Europe) UK South

Figure 3.3 – Azure resource creation region

Note from the preceding screenshot that only regions and not individual data centers can be selected when creating a virtual machine.

When creating Azure resources, regions are categorized under the headings **Recommended** or **Other**; the appropriate option should be selected for the scenario.

A **Recommended** zone is designed to support Availability Zones, now or planned for the future.

The regions categorized as **Other** refer to the data pairing of regions to provide the data residency boundaries and the low latency for disaster recovery purposes; these regions do not support *Availability Zones*.

This section looked at the concepts of Azure regions and geographies. The following section looks at Azure Edge Zones.

Azure Edge Zones

Microsoft is extending the Azure platform to incorporate *edge computing*; this allows data processing and code to be executed closer to the end user when sensitivity to network latency is critical.

Edge Zones can provide consistent Azure computing resources; the processing is completed outside of the Microsoft data center regions but within Microsoft point-of-presence locations or third-party data center locations or processed at the customer's data center locations. An organization can choose to run some resources in Azure regions and some in any of the three Azure Edge Zones based on requirements.

The following diagram positions Azure data center regions and the Azure Edge Zones:

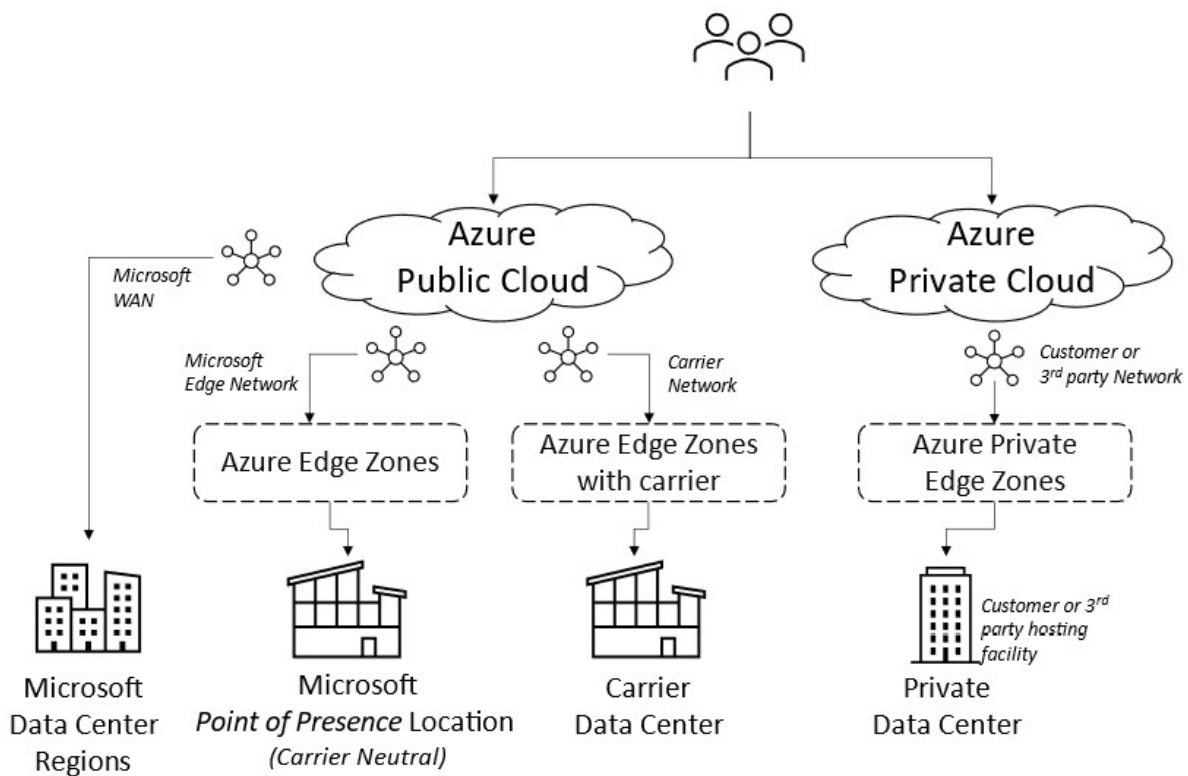


Figure 3.4 – Azure edge zones

The following describes the three edge zone scenarios:

- *Azure Edge Zones* are still Azure public cloud (*shared*) platform resources (*Microsoft hardware*) within Microsoft's *point of presence* edge locations; these edge locations typically already provide services such as **ExpressRoute**, **CDN**, and the **Azure Front Door service**. These zones are connected to Microsoft's global network.

- *Azure Edge Zones with carrier* are still Azure public cloud (*shared*) platform resources (*Microsoft hardware*) but are now shipped to exist within the carrier's data center locations; these zones are part of the carrier's global network.
- *Azure Private Edge Zone* is part of the Azure Stack (*private*) cloud platform (*customer or third-party hardware*) within the customer's location or a third-party hosting provider. These zones are part of the customer or hosting provider's private network and not part of a carrier or Microsoft network. These zones can be connected to Microsoft data center regions using a VPN or ExpressRoute to establish hybrid connectivity if required.

This section looked at the concepts of Azure Edge Zones. The following section looks at Azure Sovereign Clouds.

Azure Sovereign Clouds

Azure supports what is referred to as *Sovereign Clouds*; these support greater compliance for specific markets. These are isolated cloud platforms that run dedicated hardware and isolated networks in the regions where these Sovereign Clouds are located.

The Sovereign Cloud platforms also have their own portals with different URLs and different service endpoints in DNS. The following are some examples of the URLs and endpoints to connect to in DNS:

- Azure US Government cloud: <https://portal.azure.us> – *.azurewebsites.us
- Azure China cloud: <https://portal.azure.cn> – *.chinacloudsites.cn (*operated by 21Vianet in compliance with Chinese regulations*)
- Azure German cloud: <https://portal.microsoftazure.de> – *.azurewebsites.de

Alongside the public cloud and sovereign cloud platforms, Azure also provides the Microsoft private cloud *Azure Stack* family, which comprises three products, **Azure Stack Hub**, **Azure Stack HCI**, and **Azure Stack Edge**. The following diagram positions the different Azure cloud platforms:

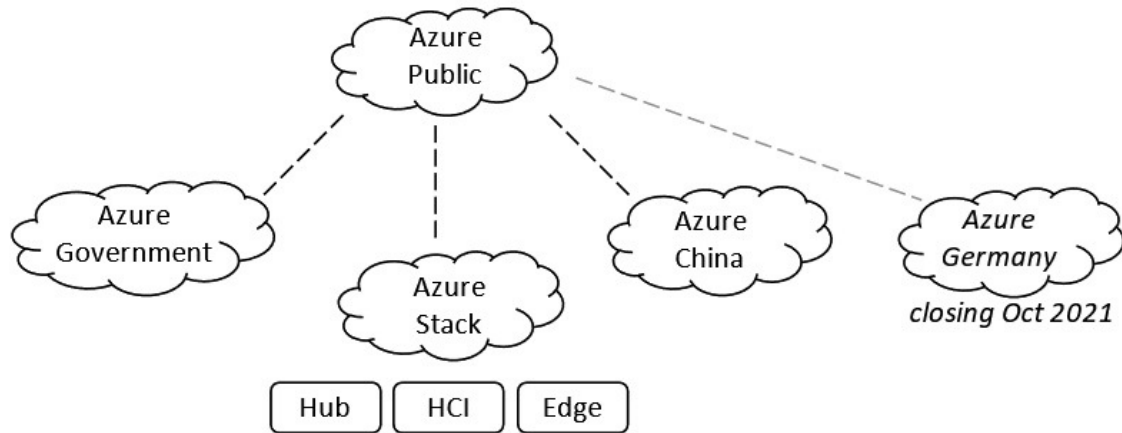


Figure 3.5 – Azure public and non-public clouds put a box around Azure Stack

Azure Stack is not provided in Microsoft regions; it is a private cloud platform. As such, it is not hosted in Microsoft data centers but hosted in an organization's facility or a third-party provider's.

This section looked at the Azure Sovereign Clouds. The following section looks at availability solutions natively provided within the Azure platform.

Availability components

When creating resources in Azure, we should think back to the cloud computing services model of **Infrastructure as a Service (IaaS)**, **Platform as a Service (PaaS)**, serverless, and **Software as a Service (SaaS)**, with the same principles and approach being applied to the responsibility of the *availability* of resources. This means we should consider how we will make data, app, compute, and other services available in a failure at the app or virtual machine level, the hardware level, the data center, or the region level.

Depending on the service, Microsoft is responsible for providing a range of availability options, but you are responsible for implementing them to ensure you have designed an availability strategy that meets your **Service Level Agreement (SLA)** requirements.

Microsoft's SLAs for its services are based on financial compensation when the defined SLA is not met. As a reference, the Microsoft SLAs for all services can be found at <https://azure.microsoft.com/en-gb/support/legal/sla/>.

The following diagram aims to summarize these components by looking at virtual machines and storage resources. You will see how you can increase the SLA and durability by using these components:

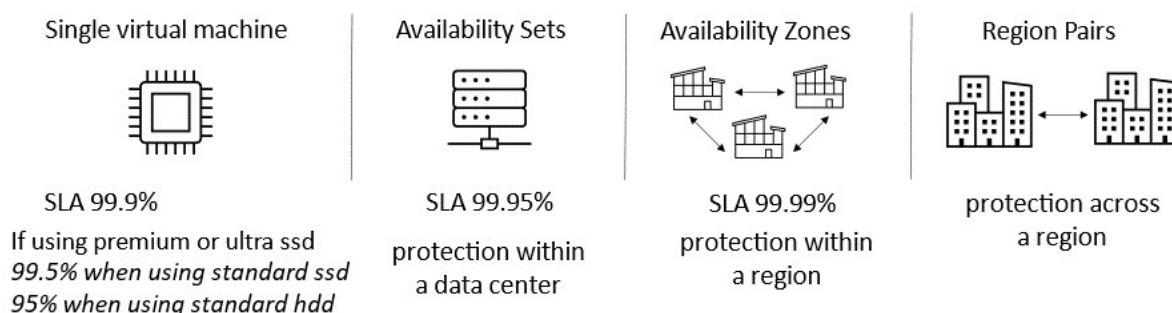


Figure 3.6 – Azure availability components

A solution will be based on the resources that are to be protected, the nature of the failure to protect the resources from, as well as considering the SLA requirements.

The following are the availability components that Azure can provide to build a solution. These components can be considered as building blocks to use as required:

- **Within a data center component:**

Example failure that could occur: Hardware rack failure; power, network.

Availability component: Availability set in the case of a virtual machine or **Locally Redundant Storage (LRS)** in the case of storage.

- **Within a region component:**

Example failure that could occur: Data center failure; power, network, cooling.

Availability component: Availability Zone in the case of a virtual machine or **Zone Redundant Storage (ZRS)** in the case of storage; synchronous replication is used.

- **Across a region component:**

Example failure that could occur: Entire region failure – multiple data centers in that region that suffer failures such as power, network, or cooling.

Availability component: **Azure Site Recovery (ASR)** in the case of a virtual machine or **Geo-Redundant Storage (GRS)** in the case of storage; asynchronous replication is used.

In this section, we introduced the availability components of the native Azure platform. In the following section, we look at adopting a risk model.

Adopting a risk model

Much like in the security area, where we take a defense-in-depth approach, we should consider taking a similar approach to implementing *availability* into a solution. We should understand the possible failures at the level we are concerned with, determine the impact and probability, and then implement the most appropriate solution; this should be driven by determining your required SLA, that is, the uptime required for a resource. The SLA of a resource can be increased by combining different availability components. We look at SLAs in more detail in [Chapter 12, Azure Service-Level Agreements](#).

The following diagram outlines the risk model approach:

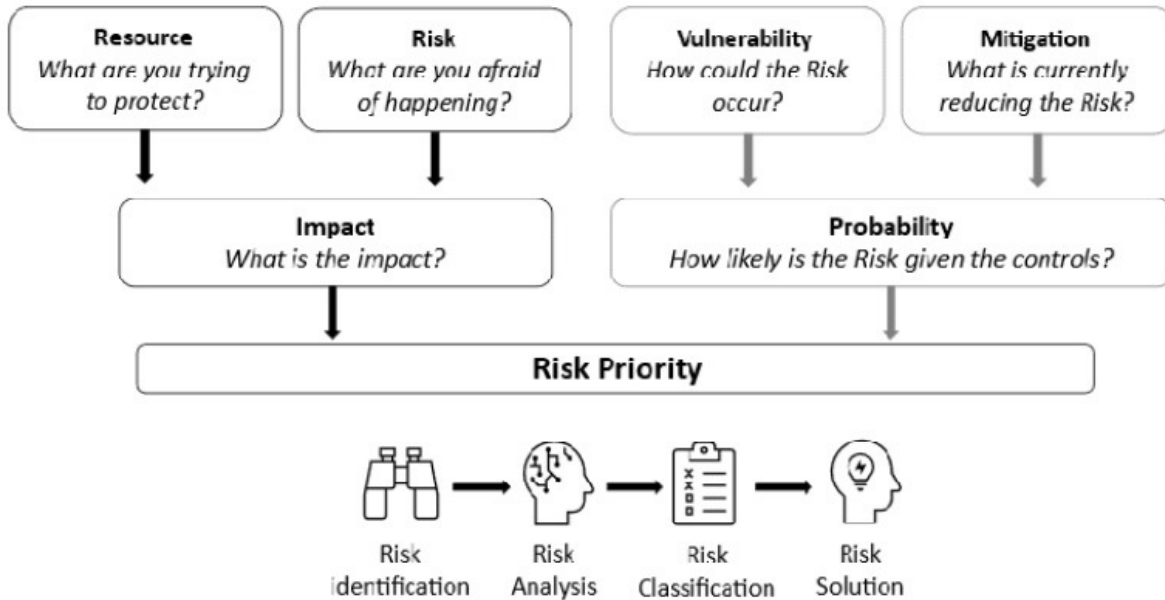


Figure 3.7 – Adopting a risk model

In this section, we looked at adopting a risk model. The following section looks at availability sets.

Availability sets

In a nutshell, **availability sets** are logical groupings of virtual machines that provide the availability of the virtual machines within a data center and provide the virtual machines a 99.95% SLA.

Availability sets deploy virtual machines across different hardware instances through what is known as a *fault domain*, meaning the virtual machines are placed on hardware that is in physically separated racks when they are created. If there is a *within-rack* failure, then all virtual machines are not impacted; only a subset will be impacted, as you have *spread your eggs over many baskets*. Without availability sets, however, you have *kept your eggs all in the same basket*.

Without virtual machines being part of an availability set, the best SLA for a single virtual machine will be 99.9% when ultra or premium SSD disks are used. This drops down to 99% when standard SSD disks are used and 95% when standard HDD is used.

These SLAs equate to the following figures:

- 99.5 % availability results in a period of allowed downtime/unavailability of 3h 39m 8s

- 99.9 % availability results in a period of allowed downtime/unavailability of 43m 49s
- 99.99 % availability results in a period of allowed downtime/unavailability of 4m 22s

The following diagram helps to visualize resource placement in an availability set:

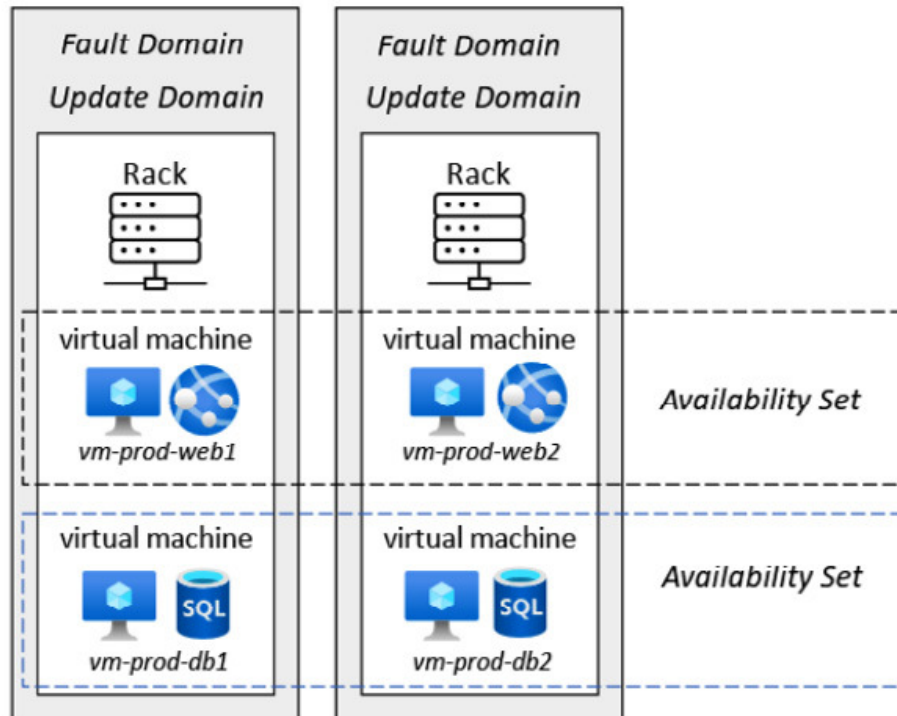


Figure 3.8 – Azure availability sets

Availability sets also act as an update (patching) boundary; an update domain is automatically assigned to all virtual machines that are part of an availability set. Updates to one set of virtual machines are isolated from other virtual machines to ensure that only one update domain gets updated at a time. If an update impacted the service of virtual machines in one update domain, that update would not be applied to other update domains.

Some points to consider for availability sets are as follows:

- Availability sets are not billable items.
- A virtual machine is part of a single availability set.
- Virtual machines can only be added to an availability set at the time of creation. It is not possible to add to an availability set or change the availability set after the virtual machines have been created; in this case, you would need to delete the virtual machine and redeploy, but this time as part of an availability set. Planning is critical for this purpose.

- You might consider deploying a single virtual machine as part of an availability set already from the start. While it won't help meet the availability set SLA – as that requires at least two instances or more – it would make your workload *ready* for the future without redeploying.
- You will need to create the virtual machine in the same resource group as the availability set to add the virtual machine.
- There are three fault domains and five update domains (*a maximum of 20 can be configured*) as part of an availability set.
- The assignment is carried out automatically and horizontally in sequence, that is, **vm1** in *fault domain #1*, **vm2** in *fault domain #2*, and **vm3** in *fault domain #3*, as we have only three fault domains in this example. This means that **vm4** will be assigned *fault domain #1*.

This section looked at availability sets. The following section takes a close look at fault domains and update domains.

Fault domain

A *fault domain* is a group of resources in a data center rack that share the same *power* and *network*. When a failure occurs in a fault domain, all resources in that fault domain are unavailable.

In the following diagram, each web server and database server is placed in a separate fault domain. A failure can occur in one of the fault domains, but only one web server and one database server will be offline; as we looked at earlier, this is the approach of *spreading your eggs across different baskets*:

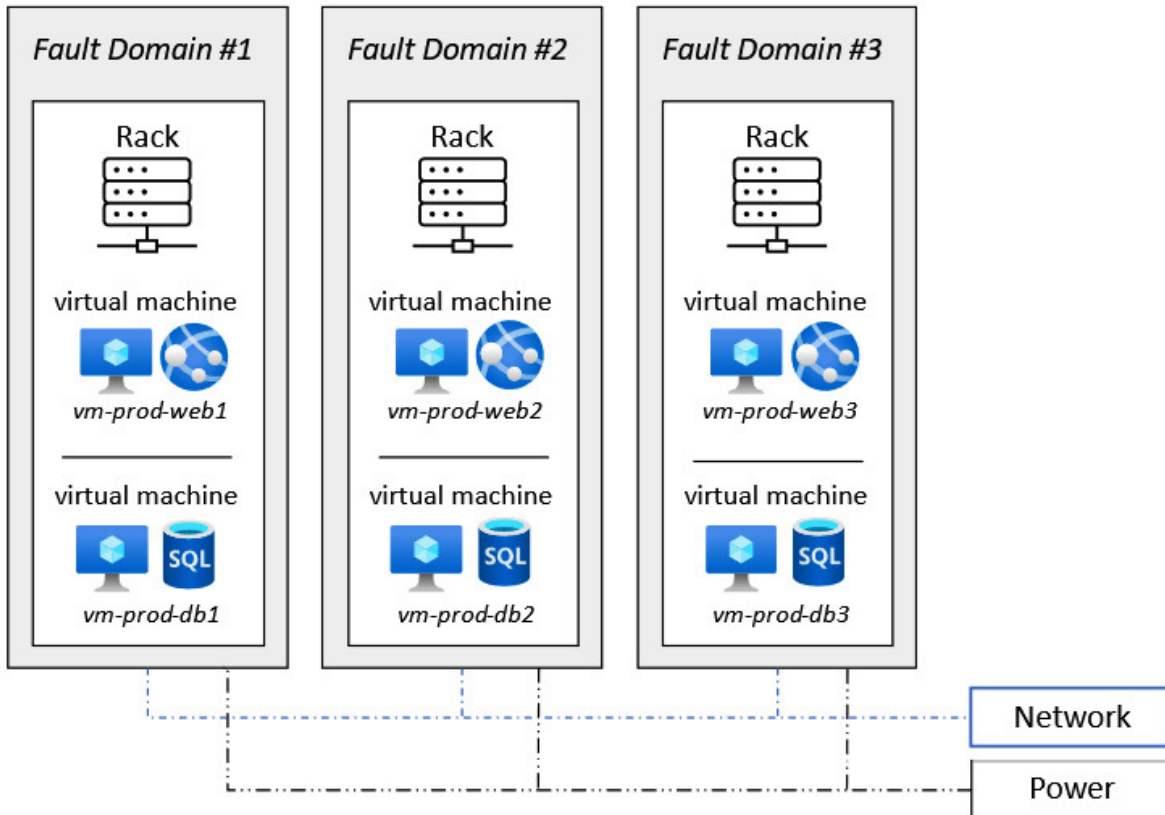


Figure 3.9 – Availability set fault domain

If the servers shared the same fault domain, then they all would be affected by the failure. This section looked at fault domains. The following section looks at update domains.

Update domain

Update domains are similar to fault domains, but they relate to patching schedules. In a nutshell, all virtual machines in the same update domain will receive the updates and be rebooted together. For this reason, do not put both web servers in the same update domain as both will be rebooted simultaneously.

You should put each web server in its own update domain, so they are rebooted independently, and so there is always a web server available to handle requests. The same approach should be taken for the database servers, so there is always a database server available to handle requests from the web servers, and so both database servers are not rebooted at the same time and are potentially both unavailable at the same time.

The following diagram helps to visualize the assignment of virtual machines across update domains:

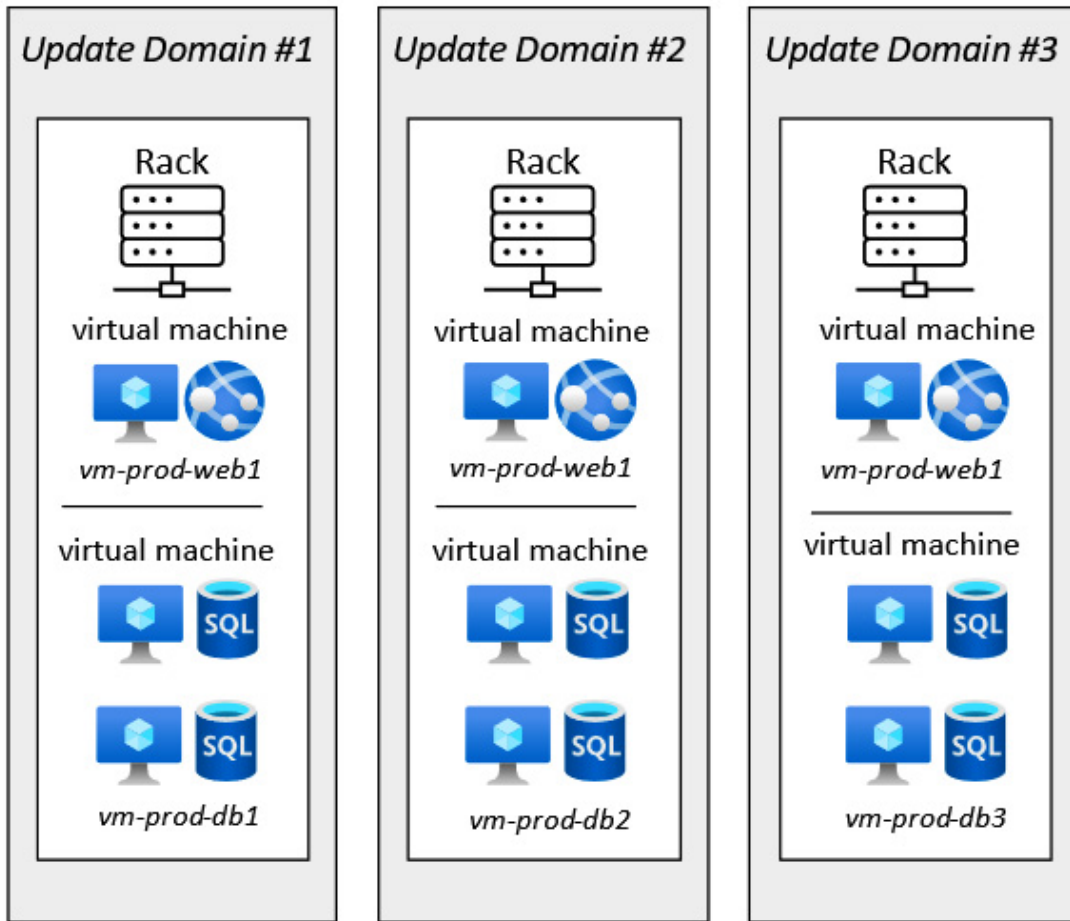


Figure 3.10 – Availability set update domain

This section looked at availability sets. The following section looks at Availability Zones.

Availability Zones

Some regions are further divided into *Availability Zones*, although they are not available in all regions, and not all resources support Availability Zones.

The purpose of an Availability Zone is to provide redundancy *within a region*, that is, to protect against a data center failure, so that a failure related to something such as power, cooling, or connectivity does not impact operations of any resources running in a single data center.

Note that Availability Zones do not protect resources from a *region* failure, only a *data center* failure; protecting a virtual machine from an *in-rack* failure, for example, *within a data center*, is the scope of *availability sets*, which we looked at in the previous section.

The following diagram outlines the Azure Availability Zones topology:

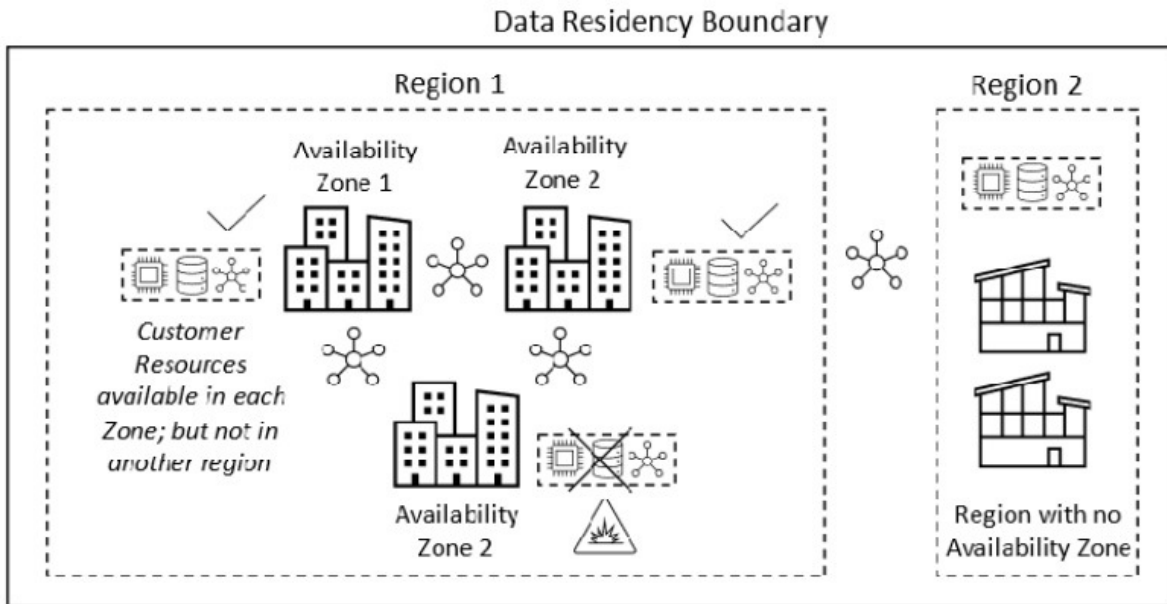


Figure 3.11 – Azure availability zones

The availability of resources running in a zone is provided by each zone being protected using independent power, cooling, and networking from other zones. A service outage in one zone will not affect the availability of resources running in different zones.

A higher-level SLA can be provided to you by Microsoft when you choose to use Availability Zones for your deployed resources for those that support this functionality and regions where this is available. In the example of providing availability for virtual machines, a 99.99% uptime is guaranteed by Microsoft if a minimum of two virtual machines is deployed into two or more zones; there is synchronous replication of the virtual machines, which is automatically taken care of by Microsoft.

Without an Availability Zone, should a data center have a failure where your resource is running, your resource will be offline until that data center is brought back online and services are restored.

Availability Zones allow resources, connectivity, and traffic to remain within the primary region but are hosted within a different physically isolated and distanced set of data center buildings.

To protect against an entire region becoming unavailable, resources can be replicated to a secondary region to protect against a service failure within the primary region where the resources are running. The issue may be that you do not want or cannot have your services failing over or running from another region; this may be for reasons such as compliance, latency, or connectivity. You may rely on VPNs or ExpressRoute that are not configured to be available in the secondary region. There may also be many other dependencies that mean having another region as a redundancy option is not viable.

Azure resources are placed into one of three categories as outlined here:

- **Zonal services:** Provide the ability to select an Availability Zone for the resources; resources can be pinned to a specific zone based on your needs, performance, or latency, for example.
- **Zone-redundant services:** The replication of resources across zones is automatic, and you cannot define the replication settings of how the resources are distributed across the zones.
- **Non-regional services:** Services are available in all geographies and are not affected by zone-wide or region-wide outages.

This section looked at Availability Zones. The following section looks at proximity placement groups.

Proximity placement groups

Proximity placement groups are a logical entity, and an architectural component that should be considered in any solution design where low latency between Azure infrastructure compute resources are required. They ensure the compute resources are physically adjacent and collocated within the same physical data center and not across data centers.

Proximity placement groups are important where latency is considered. Placing the resources closer within a single availability zone is possible, but the challenge is that an availability zone can expand to span multiple physical data centers. The web frontend tier virtual machines may be in one data center, but the database tier is in a different data center's racks; the latency impact here should be readily apparent.

Proximity placement groups alleviate this as previously virtual machine-accelerated networking was one solution to latency.

The following diagram outlines the placement of virtual machines without a proximity placement group:

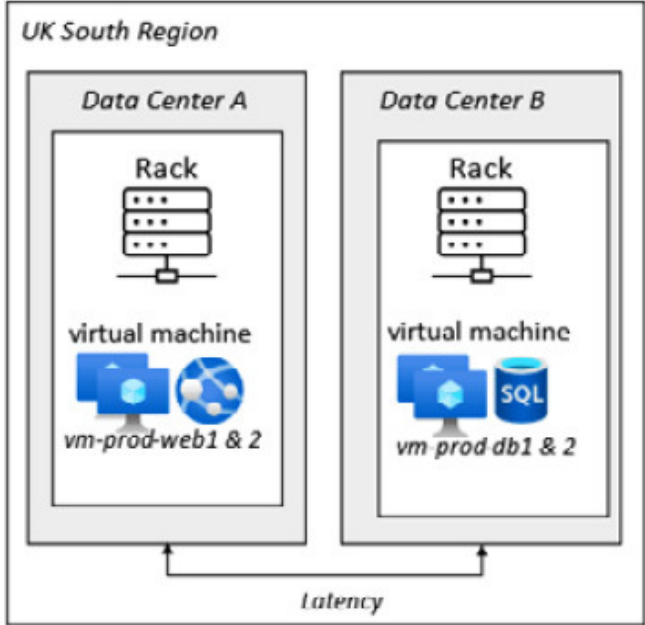


Figure 3.12 – Virtual machine placement without proximity placement groups

In the preceding diagram, we see that two virtual machines that need to communicate with each other could be placed by Microsoft in different data centers several kilometers apart, which may cause issues due to latency. The following diagram outlines the Azure proximity placement groups topology and how this can reduce latency; the virtual machines are now located within the same data center, and the latency is now reduced to within the same rack:

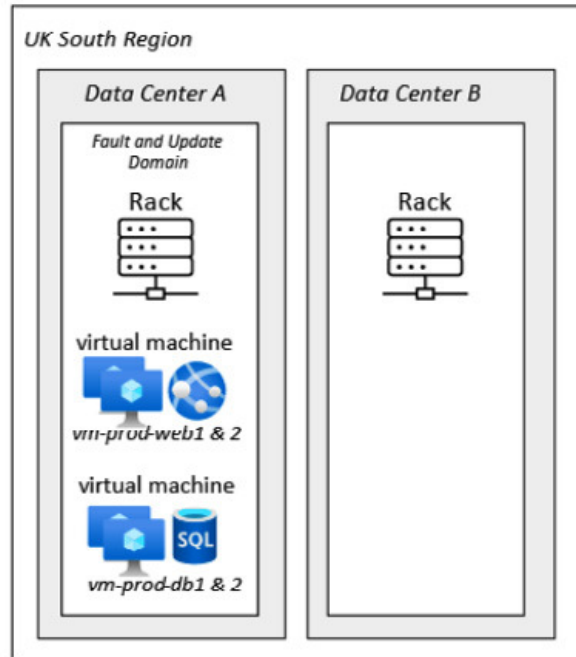


Figure 3.13 – Virtual machine placement without proximity placement groups

The preceding diagram for proximity placement groups while solving a latency problem does introduce an availability issue; as both virtual machines are now located in the same rack, they now also share the same fault and update domain. This issue can be resolved by using an availability set so that each virtual machine will be placed in a different rack and different fault and update domains. The following diagram outlines this:

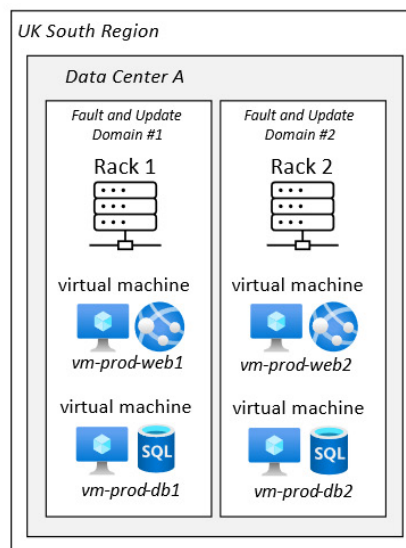


Figure 3.14 – Proximity placement groups and availability sets

The following information should be noted about proximity placement groups:

- They are Azure resources and need to be created before they can be used.
- They can be used with virtual machines, virtual machine scale sets, and availability sets.
- When creating the compute resource, you specify the proximity placement group previously created.
- You can move existing compute resources into a proximity placement group; the resource will need to be stopped (deallocated) to move.
- They are set at the resource level not the individual virtual machine level for availability sets and virtual machine scale sets.
- A workload such as SAP Hana with SAP NetWeaver is a good example to illustrate the importance of having the machines far enough away for high availability/SLA but close enough not to face latency issues in communication.

This section looked at proximity placement groups. The following section looks at ASR.

Azure Site Recovery

ASR is the Azure platform disaster recovery service for virtual machine resources. It protects against complete region failure, that is, if multiple data centers in that region fail. It is, in effect, a *data availability* solution and preserves the need for regional protection within data residency boundaries. Although not illustrated here, ASR also has a *hybrid* aspect, which allows you to build a disaster recovery scenario for on-premises physical/VMware/Hyper-V or AWS virtual machines to Azure virtual machines:

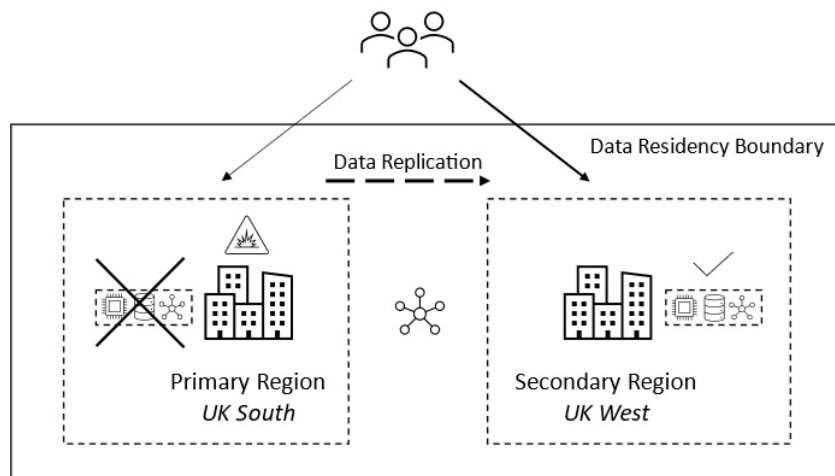


Figure 3.15 – Azure region-to-region protection

This section looked at ASR and how to protect from entire region failures. The following section looks at Azure resource management.

Azure resource management

In the previous chapter, we looked at the *digital transformation methodology*; it is this last phase of *Secure and Manage* that we turn our attention to in this section.

It is important to ensure that any workloads or data running in a cloud computing environment are managed in the same governed, controlled, secure, and protected manner that they would be for a traditional computing model. The following diagram aims to visualize these aspects:

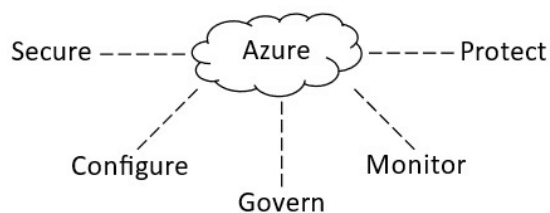


Figure 3.16 – Azure resource management

These areas to consider should include the following:

- *Availability* through redundancy, replication, and traffic management
- *Protection* through backup and disaster recovery
- *Security* through threat protection and security posture management
- *Configuration* through automation, scripting, and update management
- *Governance* through access control, compliance, and cost management
- *Monitoring* through the collection of security incidents, events, resource health, performance metrics, logs, and diagnostics

The foundation for all management within Azure is ARM, which is an **Application Programming Interface (API)** and is the management and deployment service for Azure.

All user and resource interactions pass through the ARM API; you can think of it as the brain or engine of Azure. It provides a single consistent management layer and orchestration engine processing all requests for all interactions with the Azure platform, whether via the portal, command-line interfaces, templates, Microsoft utilities, and third-party software; each of these calls the ARM API.

Governance and planning is an essential stage when creating resources in Azure. It can prevent re-work and potentially tearing down aspects that may have been implemented without fully appreciating what should be considered, understanding all options and their benefits or limitations on the desired outcome. The first aspects of planning are billing control and access control, which will shape the strategy and approach for implementing subscriptions, management groups, and resource groups. This section introduced the concepts of Azure resource management. The following section looks at Azure management scopes.

Azure management scopes

Azure provides four management scopes, listed as follows so that RBAC and **Azure Policy** can be targeted at those levels:

- *Management group level*
- *Subscription level*
- *Resources group level*
- *Resource level*

The following diagram outlines these four management scopes and the hierarchy:

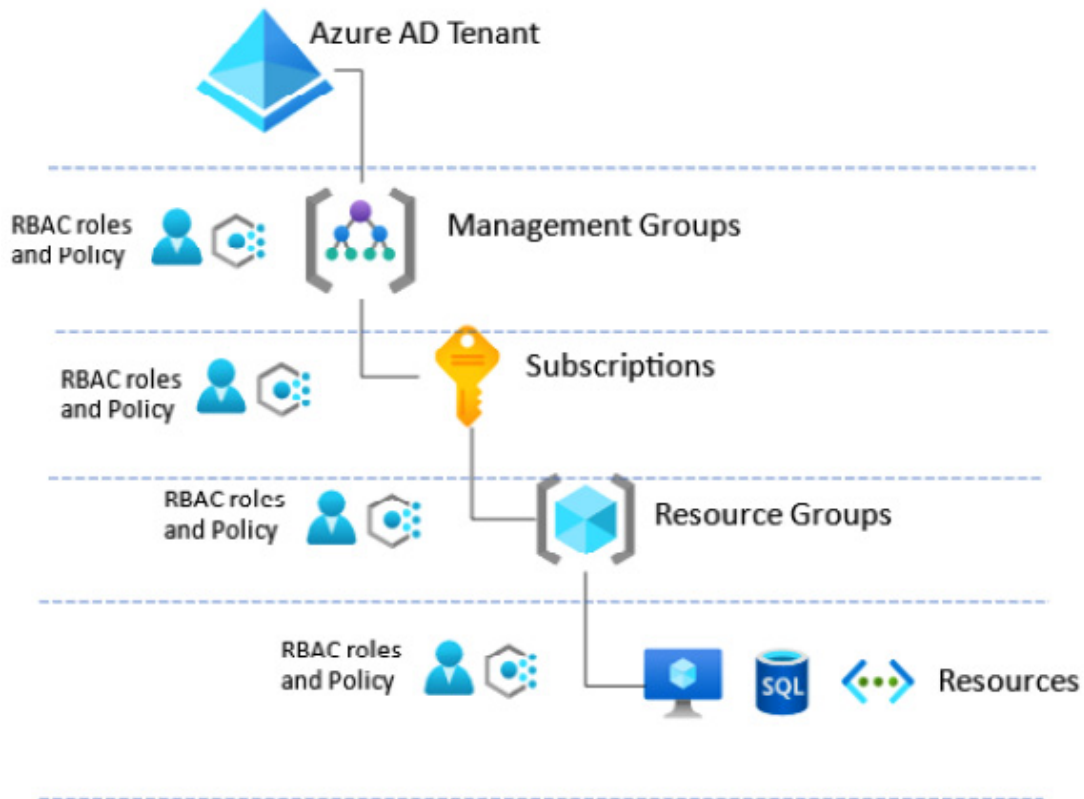


Figure 3.17 – Azure management scopes

This section introduced the concepts of Azure management scopes, which are the levels at which access and policy can be applied. The following section looks at Azure management groups.

Azure management groups

Azure management groups are logical containers that group together Azure subscriptions and can be considered a governance and management layer to implement access control and policies.

Management groups are the most effective governance scope when there are multiple Azure subscriptions in a tenant.

There are some limitations of management groups to be aware of, however:

- Management groups can only contain other management groups and subscriptions; they cannot include resource groups or resources.
- The scope is per tenant and not across tenants.
- The parent management group cannot be deleted or moved; it can, though, be renamed.
- Only one parent management group is supported, and only one parent hierarchy is a single tenant.
- Can support many children management groups, up to 1,000.
- A hierarchy of up to six levels is supported.

This hierarchy may be applied across business units, regions, or environments, wherever you may need to define billing or access control boundaries. In this scenario, planning is everything.

The following diagram shows just one example of the hierarchy and the relationships between the different subscription types within a business:

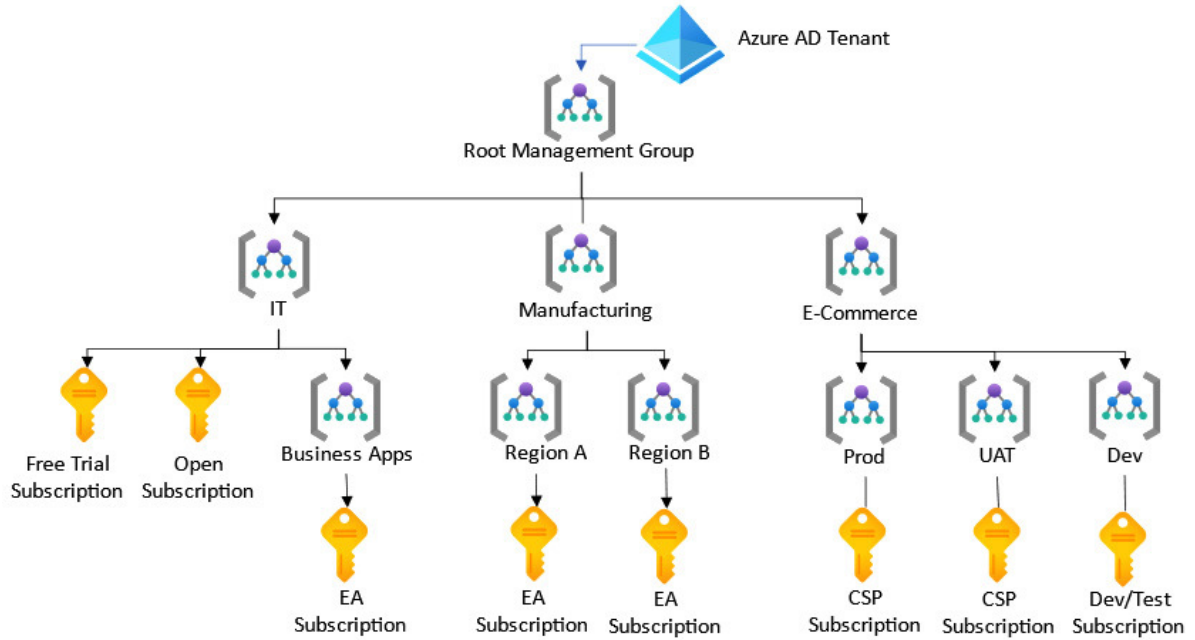


Figure 3.18 – Azure management group and subscription relationships

This section looked at Azure management groups and the relationship to Azure subscriptions. The following section looks at Azure subscriptions.

Azure subscriptions

Azure subscriptions can be considered the *logical containers* of Azure resources that share the same *billing boundary* and an *access control boundary*.

An Azure subscription is a billing mechanism for the Azure resources you consume within a tenancy. It can be likened to a bar tab, a method of paying for everything you have consumed all together and itemized at the end of the night without paying individually each time you consume a drink. When a bar tab is opened, a payment method must be provided, such as a credit card, so that there is one bill to settle for everything put on the tab at the end of the night; you can think of the subscription as the tab.

Every time a resource is created within Azure, an Azure subscription must be selected against which to associate the consumption of resources (*or tab in our previous example*). This means the Azure subscription(s) would be created first to be able to create any resources.

At the end of the month, all the Azure resources created in that subscription will be billed in a single itemized invoice.

The following diagram outlines Azure subscriptions:

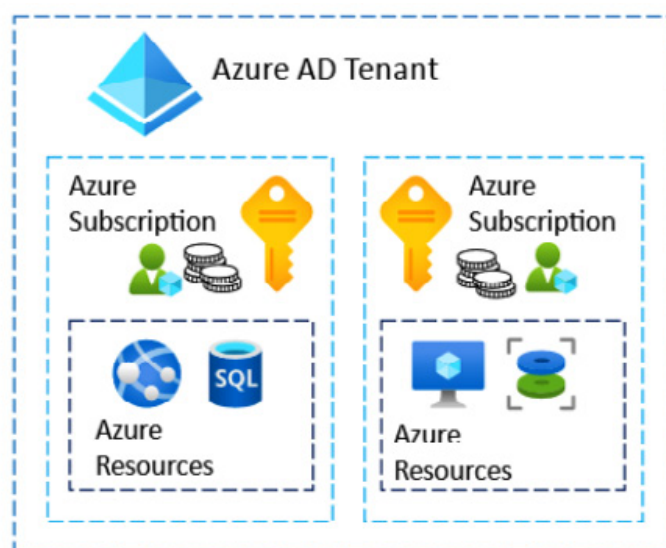


Figure 3.19 – Azure subscriptions

The Azure subscription will need to be created in an existing Azure AD tenancy or a *new* Azure AD tenancy; this is the foundation and starting point. *Management groups* are then created if required, and finally, *resource groups* are planned as needed. We will cover management groups and resource groups in a later section of this chapter.

Subscriptions cannot be merged, but resources can be moved into another subscription and delete the subscription or allow it to expire; any resources in the subscription are deleted when the subscription is deleted. However, when a subscription expires, the resources remain. In either case, the tenancy remains in place; this is not removed.

The billing owner of the subscription can also be changed, much like a bar tab could be run up and transferred to somebody else to take it over and settle the tab.

When planning to start creating resources in Azure, you must first define access and billing control; this will shape the subscription and resource group strategy. You should create the tenancy, subscription, and resource groups in that order.

Planning is to be considered for your subscriptions and whether you will have multiple subscriptions within a tenancy. You may wish to have one for production resources and one for development and testing purposes to isolate access to the resources through the subscription or just to control who will receive the invoice for any resources created within each subscription. This will depend on your needs, and we look at creating additional subscriptions later in this chapter.

This section introduced the concept of Azure subscriptions and how they are a billing mechanism, as well as a billing and access control boundary.

The following section looks at the relationships between subscriptions and tenants.

The relationship between tenants and subscriptions

As we saw in the last section, there are certain relationships between *subscriptions* and *tenants*. The relationships also extend to other entities such as *resource groups* and the Azure *resources* themselves. There are also relationships between *users*, *groups*, and *apps* such as a Microsoft 365 subscription. *Resource groups* are logical containers for resources; these will be covered in more detail later in this chapter.

The following diagram outlines the relationship between these entities:

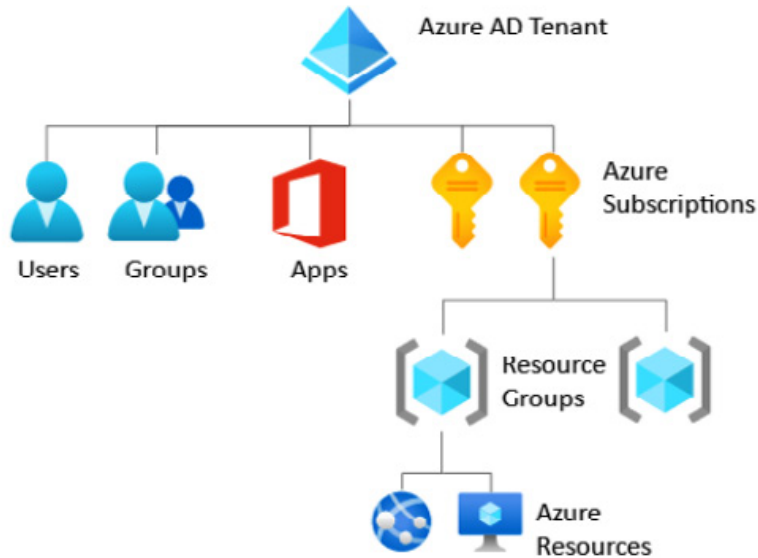


Figure 3.20 – Azure subscription and tenant relationship

In a nutshell, an Azure AD tenant contains users, groups, Microsoft 365 subscriptions, and Azure subscriptions. Azure subscriptions contain resource groups, which contain resources. The following facts should be observed concerning these relationships:

- Each *subscription* in Azure can be associated with *only one* Azure AD tenant, in a parent and child relationship, with the Azure AD tenant being the parent.
- A *subscription* can be moved to a different Azure tenant.
- The subscription billing owner can be changed, such as moving from being invoiced by Microsoft to being invoiced by a **Cloud Solution Provider (CSP)**.
- Each *subscription* can have multiple resource groups; each resource group can be part of *only one* subscription.
- Each *resource* can be associated with *only one* resource group; in a parent and child relationship, the resource group is the parent.
- Each *user* can be associated with *only one* Azure AD tenant, with the Azure AD tenant being the parent.
- Each *user* can access *more than one* Azure AD tenant.
- Each *group* can be associated with *only one* Azure AD tenant, with the Azure AD tenant being the parent.
- Each *Microsoft 365* subscription can be associated with *only one* Azure AD tenant (a `<name.onmicrosoft.com>` domain), with the Azure AD tenant being the parent.

This section looked at the relationships between subscriptions and tenants and the broader relationships between resource groups, resources, users, groups, and other subscriptions such as Microsoft 365. The following section looks at Azure subscription management.

Azure subscription access control

It is essential to control access to subscriptions to ensure governance of creating resources within a subscription (*who is adding drinks to the tab from our previous section's analogy*); this utilizes *RBAC*.

RBAC provides the ability to assign multiple accounts different access levels to a subscription as needed by an organization. Multiple owners are set for a subscription, although the number of owners should be limited to a maximum of three for best operational security practices as defined in Azure Security Center. Azure AD accounts are primarily used to assign access to Azure subscriptions through role assignments.

We will cover RBAC in more detail later in this book; we will cover the basic concepts only in this section.

There are two levels of access control, **authentication** (often shortened to **AuthN**) and **authorization** (often shortened to **AuthZ**):

- *Authentication* is proving who you say you are.
- *Authorization* is what actions you are allowed to do as that identity, such as read data, modify data, delete data, or give others access to data.

In terms of managing access to an Azure subscription, *authentication* comes from *Azure AD* and *authorization* comes from *Azure RBAC roles*.

The following diagram shows the span between roles:

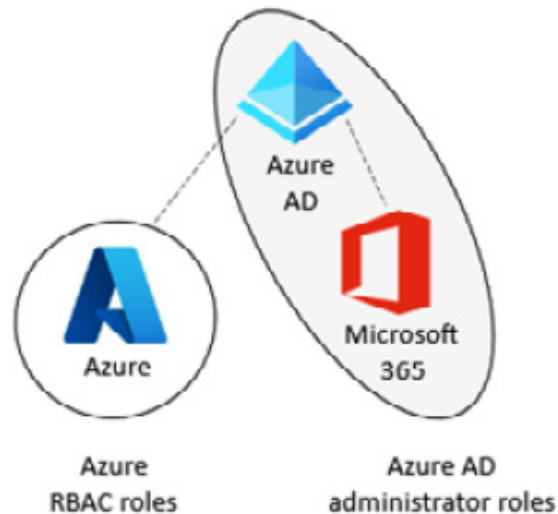


Figure 3.21 – Azure subscription access control

The following information should be noted from the previous diagram:

- By default, the Azure roles and Azure AD are isolated and do not span or overlap from Azure AD, as in Microsoft 365 roles.
- Being assigned an Azure AD role gives no assignment of any Azure roles.
- By default, there is no access to Azure resources with an account with the Global Administrator role.
- Access to Azure resources must be explicitly allowed and can only be granted with an account that has the Owner role for a subscription

This section introduced the concepts of access control to Azure subscriptions. The following section looks at the Azure roles.

Azure roles

We look at the fine-grain roles in this section from the ARM deployment model that replaced **Azure Service Manager (ASM)**, now referred to as the *classic* model. In addition, custom roles can now be defined within the ARM model.

The four core ARM roles are as follows:

- **Owner:** This role has full access to all resources and can delegate access to others.
- **Contributor:** This role has full access to all resources but cannot delegate access to others.
- **Reader:** This role has only read access to all resources.
- **User Access Administrator:** This role can only manage user access to all resources.

The other 70+ roles are specific to individual Azure resources and give granular resource context-specific allow actions, as follows:

- **Backup Operator:** Allows management of backup services but does not allow the creation of vaults, removing backups, or giving access to others
- **Storage Account Contributor:** Allows management of storage accounts; provides access to the account key
- **Virtual Machine Administrator Login:** Allows viewing virtual machines in the portal and logging in as the administrator but does not allow virtual machine creation or disk and network management

These roles can be further customized to any individual permission required, combining a few of the 1,000 possible roles.

The roles and their assignments are managed via the **Access Control (IAM)** page for the subscription within the Azure portal.

This section looked at the Azure roles for access management. The following section looks at user access when a user is not part of the tenant's Azure AD.

External identity access

Azure AD **Business to Business (B2B)** is supported for inviting guest access to an Azure AD tenant; this is where an Azure AD account from another tenancy can be given access to the subscription within a different tenancy. In the case of **Managed Security Services Provider (MSSPs)**, **Azure Lighthouse** can also be used. Although this is not within the scope of the exam objectives, we are mentioning it here for completeness. Azure Lighthouse provides delegated admin access at the subscription or resource group level to an organization's tenant from an MSP's managing tenant. The IAM and role assignment functionality and Azure Lighthouse will be covered in more detail in [Chapter 6, Azure Management Tools](#), of this book.

This section introduced IAM functionality based on the principle of RBAC. The following section looks at creating additional subscriptions for a tenancy.

Additional subscriptions

An organization is not limited to a single subscription; resources can be consumed from multiple subscriptions within the same tenant or another tenant the organization

owns. Any additional subscriptions can be created for access control management or billing management purposes as a subscription acts as a billing and resource management boundary.

There are subscription soft quota limits; this means that a service request must be made to Microsoft support to increase resource quota limits. Typically, this request would be for a virtual machine core count exceeded. To set the number of virtual machine cores needed, the subscription quota would need to be increased.

A subscription can be a boundary of *access control*. A subscription could be created for each different environment, such as creating a subscription for **User Acceptance Testing (UAT)**, production, or staging. Alternatively, a subscription could be created per team or project as shown in *Figure 3.22* in this section. This approach ensures control of access to resources, with individuals or groups only having the least amount of access they need to the minimum amount of resources. This is one of the core principles of governance and compliance, and through RBAC, the approach and principle of *least privilege* can be adopted. The subscription can also be used as a layer at which to enforce *Azure Policy*.

In addition to being used as a boundary of access control, a subscription can also be used as a boundary of *billing control*; this means costs incurred from Azure resources created can be allocated with a charge-back or show-back approach. With the resources being consumed being billed to that particular environment, team, or project, an organization can see precisely what environment, team, or project is costing the organization without all resources being aggregated into one subscription. **Tags** can be used to break down resources within a subscription but are out of scope for this section and covered later in this book.

A subscription is associated with only one tenant; the parent tenants control its access to Azure AD. Resources must also be associated with a subscription, which acts as the billing mechanism and billing boundary. The following diagram outlines the relationships between subscriptions, tenants, and resources:

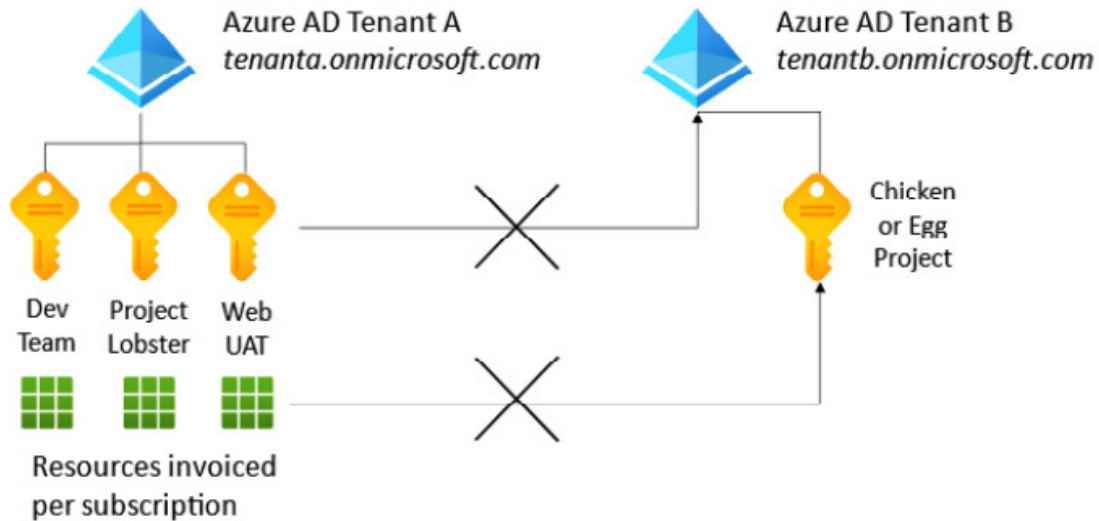


Figure 3.22 – Additional Azure subscriptions

Looking back to the bar tab analogy from the previous section, each resource can only be billed to one bar tab, although resources can be moved between subscriptions so that another subscription can be used to pick up the bill (*their own bar tab*) for those resources consumed. The scenario here could be that for a particular reason that may be appropriate, some resources that you no longer wish to be billed for under the existing resources subscription, in this case, the resource, can be moved to another subscription that is maybe more appropriate to pay the bill for those consumed resources. The resource move is a simple operation that can be done within the Azure portal.

We can see from the example shown in *Figure 3.22* that the three subscriptions associated with **Tenant A** can't be associated with **Tenant B** simultaneously; they can only have their own parent owner. Likewise, with resources, these resources can only exist within one subscription, which acts as the billing mechanism for the consumption of those resources. The resources cannot be associated with another subscription within another tenancy simultaneously. They can move owners from subscription to subscription within the same tenancy and from the parent subscription to a *different* subscription in a *different* tenancy.

This section looked at the reasons and benefits of creating additional Azure subscriptions to a tenancy. The following section takes a look at ARM.

Azure Resource Manager

In a nutshell, **ARM** is the deployment and management service for Azure; it provides a consistent management and control layer to create, update, delete, control access, and apply policy, governance, and compliance controls.

The following diagram outlines the ARM architecture:

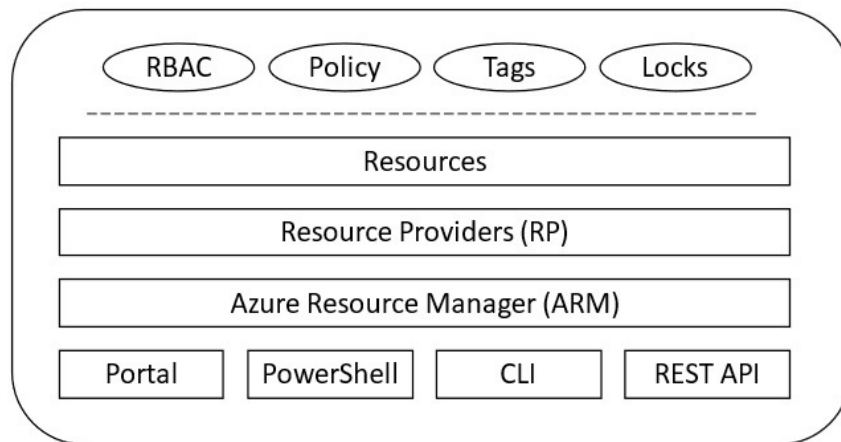


Figure 3.23 – ARM architecture

ARM provides the following functionality:

- Deployment, management, and monitoring as resource grouping rather than individual resources.
- Apply access control and policies at the resource group level, inherited by all resources in that group.
- Apply tags to resources for billing and logical grouping and management.
- Repeatable life cycle deployments, resulting in a consistent state.
- Use of deployment template files through **JavaScript Object Notation (JSON)** to define the deployment of resources.
- Creation of resource dependencies.

As its name implies, ARM takes a resource-centric approach, with every Azure object or entity being classed as a resource.

The main component of the logical architecture is the *resource group*; this is a logical container for the grouping of resources that typically share the same life cycle. The

resource group can then be a target for applying access control and policies instead of the individual resources.

The following are some terms to understand when working with ARM:

- **Resource:** In a nutshell, this is a billable item that is consumed, which is the lowest level element that can be broken down in Azure. These can be thought of as the cells, molecules, or building blocks of all services and solutions for the Azure platform. Billing is at the resource level, and each resource will have an ID with a billing meter that calculates the amount consumed and the rate to be billed. These are not billed for AD tenants, subscriptions, management groups, resource groups, and so on. These are logical entities and have no billing meters.
- **Resource group:** A logical container for resources. Resources that are to be managed together as a life cycle are grouped to target access control and compliance through policies. Resources are grouped into resource groups in the most appropriate way for management purposes.
- **Resource provider:** These are not logical, but the actual service contains and provides the Azure resources. You could consider a book as a reading resource and the library as the provider.
- **Resource Manager templates:** Files that use JSON that provide a repeatable and automated deployment methodology. These form the background of an **Infrastructure as Code (IAC)** methodology and use a *declarative* syntax instead of an *imperative* syntax, which is the approach with PowerShell.

This section introduced the concept of ARM. The following section looks at ARM templates.

ARM templates

ARM templates are an *IAC* approach to resource deployment. This approach allows the repeatable and reliable deployment of multiple dependent resources into an Azure subscription in an automated and governed manner.

ARM templates are *JSON* files (*a minimal, readable format for structuring data as an alternative to Extensible Markup Language*) that define the configuration to be used for resource deployment. These templates utilize a *declarative* syntax (as opposed to an *imperative* syntax), which means you define the desired outcome, with all the resources you want to be created and any properties you wish specified in a single request. This is instead of issuing multiple commands at each stage of the process to create each resource individually, and where there is no orchestration, meaning you must also consider dependencies and in which order commands are processed.

The following diagram shows the approach of ARM versus non-ARM IAC:

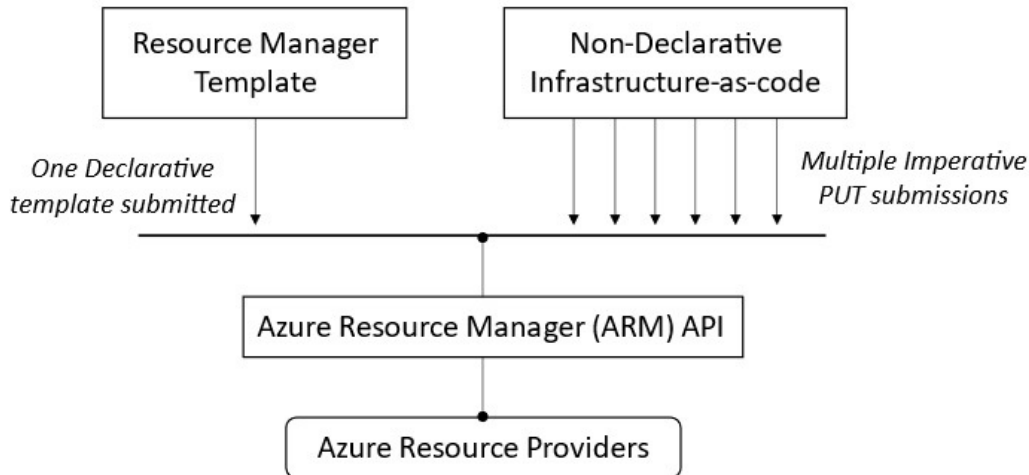


Figure 3.24 – ARM template declarative approach

ARM templates can be edited directly in the Azure portal, with a simple text editor such as Notepad, Notepad++, or with something a little more advanced such as Visual Studio Code. Whatever is defined in the template is converted to *REST API operations* for the *resource providers*.

You don't have to create ARM templates from scratch; Microsoft provides some quick-start examples that can be modified for your needs.

Providing detail on ARM templates is beyond the scope of this book; the topic is introduced here so that further study may be continued from the following URL: <https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/>.

As a closing note for this section (*again beyond the scope of this book, but mentioned for completeness*), you will also see references to **Bicep**, a new language for developing ARM templates. It provides the same capabilities as JSON templates but has a simpler syntax.

This section looked at ARM templates. The following section looks at resource groups, one of the core logical architectural components.

Resource groups

Resource groups are one of the core foundational elements of the Azure platform and ARM; the Azure platform is built around the principle of being resource-centric.

Resource groups are logical containers for Azure resources and are used as a management scope for access management and policy. We can logically group

resources in a way that means something to an organization; it may be by resource type, environment, business unit, project, or location.

The relationship between a resource group, subscription, and the resource itself is outlined in the following diagram:

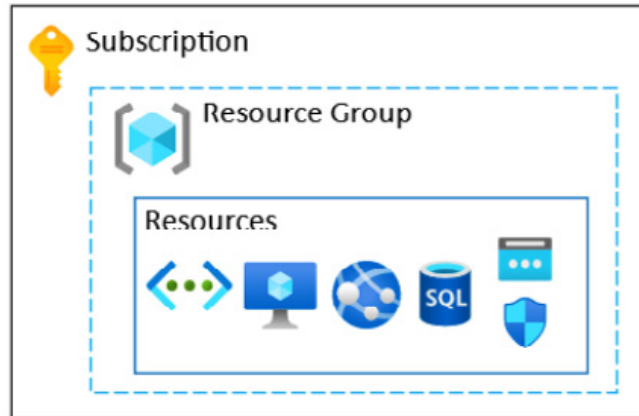


Figure 3.25 – Azure resource group relationships

This section introduced the concept of resource groups. The following section looks at some of the characteristics of resource groups.

Resource group characteristics

You should understand the following characteristics of resource groups:

- Resources must belong in a resource group and can only exist in one resource group but can be moved between resource groups.
- Resources can interact with other resources in the same resource group, other resource groups, and other subscriptions. Resources work at the *data plane* level, while resource groups and subscriptions work at the *management plane* level.
- Resource groups don't have to use the same region; they can contain resources from other regions.
- Resource groups don't contain subscriptions, but subscriptions contain resource groups.
- Resource groups are not physical; they are a logical entity and not a billable item.
- Resource groups contain metadata about the resources they include.
- Resources inherit all permissions set at the resource group level they belong to by default.
- When adding new resources to a resource group, they inherit those permissions and any access assignments.

- When moving resources, they lose the permissions of the resource group they belonged to and inherit those of the new resource group they are moved to.
- If access and permissions are assigned at the resource group level, all resources in that resource group can be managed.
- Deleting a resource group will remove all resources within that resource group and not delete the subscription or tenant.
- Because all resources are contained in the same resource group, it is easy to take action on all resources with a single activity; all resources within the resource group inherit the access assignments and policies set at the resource group.
- When assigning tags to a resource group, the resources in that resource group do not inherit those tags; you would have to apply the tags individually to each resource in that group.

This section looked at the characteristics of resource groups. The following section looks at resource group logical organization.

Azure resource group organization

As we saw in a previous section where we introduced Azure subscriptions, we noted that you must first define access and billing control when planning to create Azure resources. This will shape the subscription and resource group strategy; the order in which these should be created is the tenancy, then the subscription, then any management groups, and finally resource groups. Resource group organization is entirely subjective; it all depends on the organization's decision on how resource groups will be used. Some may like to see *location* grouping and some resource groups as *resource type* groupings, such as networks or virtual machines. This is visualized in *Figure 3.26* in this section.

Creating the required resource group requires planning to find the optimal way to organize all of the Azure resources logically; this will become increasingly important and critical as the environments grow to hundreds or thousands of resources. It is best to plan a strategy for both the logical grouping of resources and a naming convention to make identifying resources easier for management purposes.

The resource group is typically used to group all resources that share the same life cycle or have some form of dependency or the same access control and management requirements. The resources could be grouped by environments, by business unit, by

region, or by any combination that seems an appropriate approach. The following diagram outlines the different methods that could be taken:

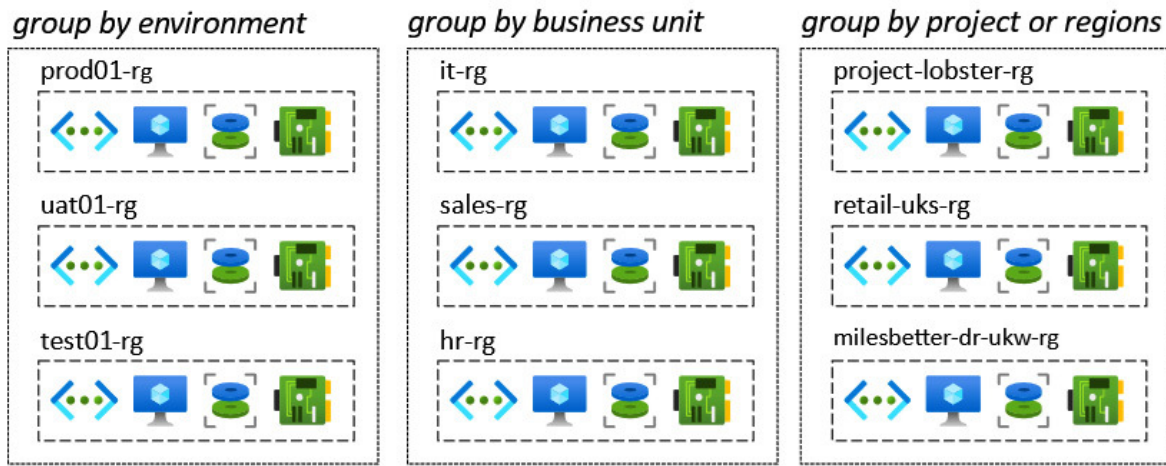


Figure 3.26 – Azure resource group organization

Individual resources can be deleted; the resource group itself can also be deleted, which will delete all the resources it contains. This is useful if there are many resources in the resource group that are no longer needed. This can also be very dangerous, as there may be resources in a resource group that are required to remain and are not to be deleted or did not mean to be deleted, oops!

There are governance controls that can be put in place, such as removing the ability to delete resource groups through RBAC, but locks can also prevent deletion. Locks will be covered in [Chapter 9, Azure Governance](#), of this book.

This section introduced Azure resource group organization and planning. The following section provides some hands-on exercises to reinforce your knowledge and increase your skills.

Hands-on exercises

To support your learning with some practical skills, we will look at the hands-on creation of some of the resources covered in this chapter.

The following resources will be created:

- Exercise 1 – Azure management groups
- Exercise 2 – Azure access assignment
- Exercise 3 – Resource groups
- Exercise 4 – Proximity placement groups
- Exercise 5 – Availability sets

Getting started

To get started with these hands-on exercises, you can create a free Azure account from this URL: <https://azure.microsoft.com/free/>.

This free Azure account provides the following:

- 12 months of free services
- \$200 credit to explore Azure for 30 days
- 25+ services that are always free

Exercise 1 – Azure management groups

In this section, we will look at the steps to create and configure management groups.

To recap from a previous section, *Azure management groups*, where management groups were covered, management groups are logical containers that group Azure subscriptions; these can be considered a governance and management layer to target access control and policies.

IMPORTANT NOTE

In contrast to resource groups, which are a flat structure with no hierarchy or nested resource groups, management groups are implemented based on a hierarchy. They have a nested, that is, root and branch, or parent and child relationship.

Follow these steps to create a management group:

1. Log in to the Azure portal at <https://portal.azure.com>. You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type in **management groups**; click on **Management groups** from the list of services shown:

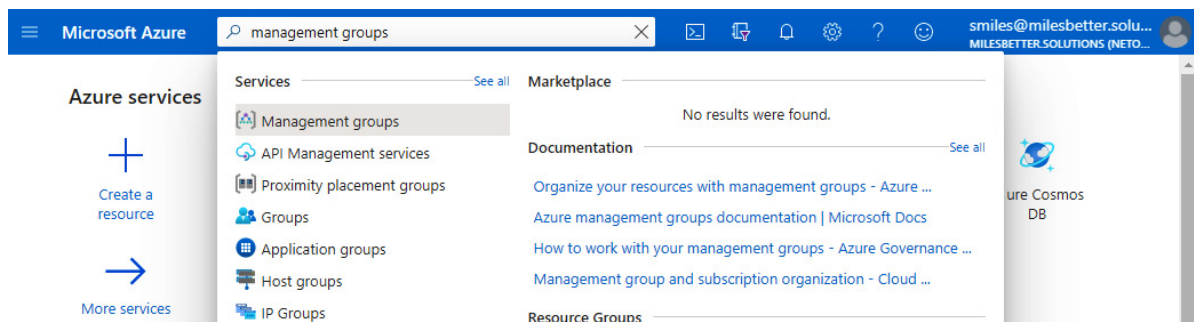


Figure 3.27 – Searching for management groups

3. In the **Management groups** blade, click on the **Start using management groups** button:

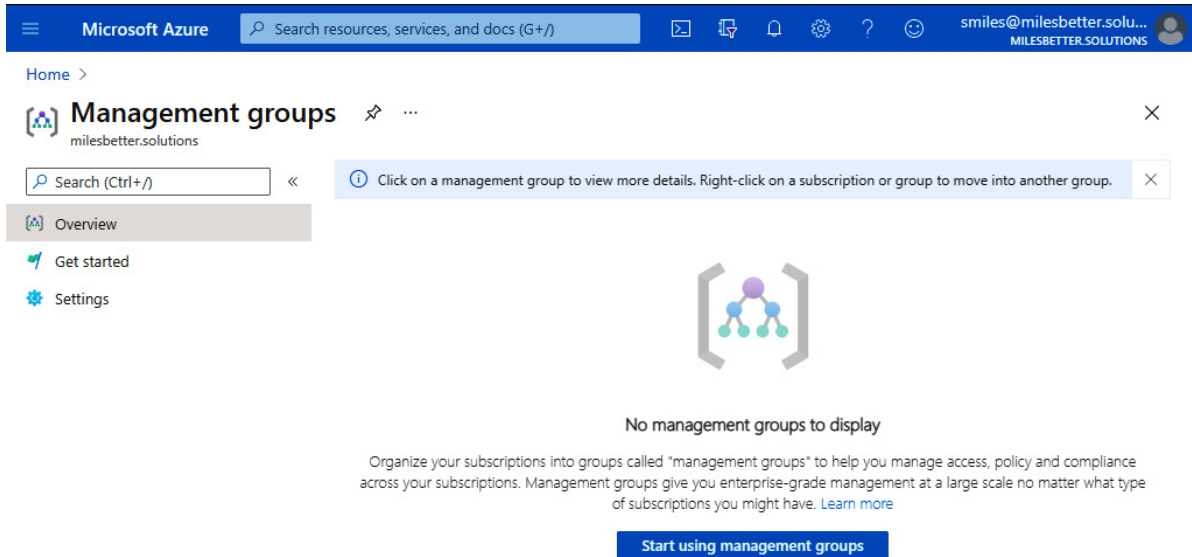


Figure 3.28 – Start using management groups

4. In the **Add management group** blade, you will need to enter a management group ID; see the following note and step on creating an ID:

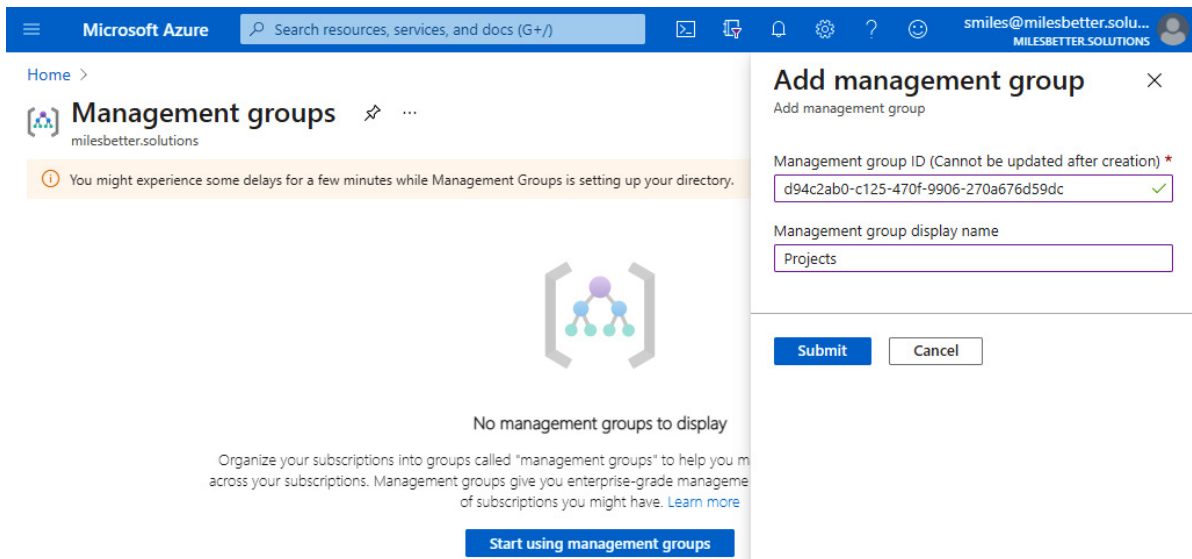


Figure 3.29 – Add management group

IMPORTANT NOTE

The management group ID is often set the same as the display name you wish to use; although, a better practice is to use an actual GUID that you self-generate. How to create this GUID is covered in the next step.

5. From the top ribbon of the Azure portal, open Cloud Shell and, using PowerShell, run the following command to create a unique GUID to use:

8. The new management group (called **Projects**) can be seen in the hierarchy from the **Management groups** blade:

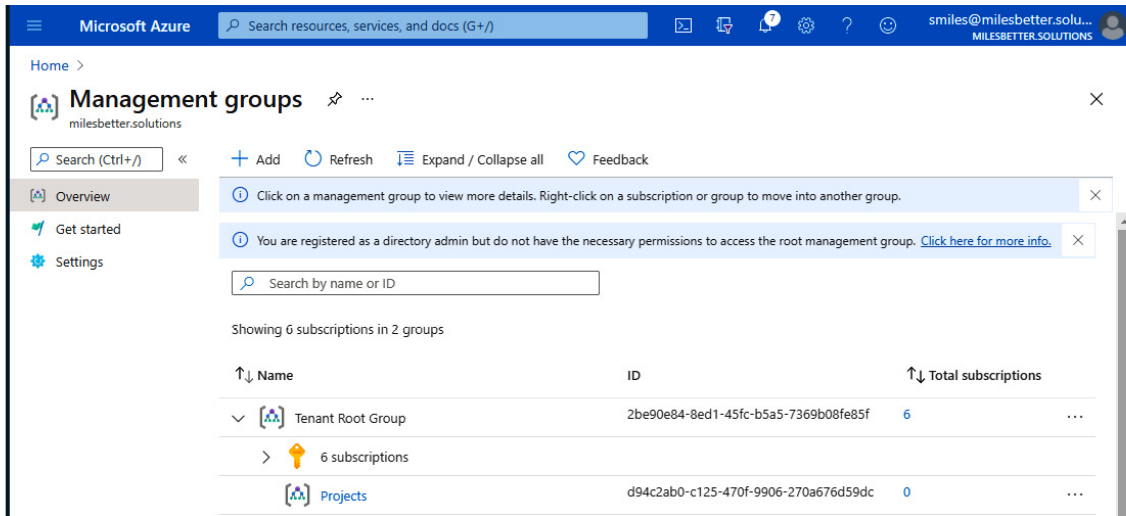


Figure 3.32 – Review management groups created

9. To move subscriptions into the management group, right-click the subscription to move and select **-> Move**:

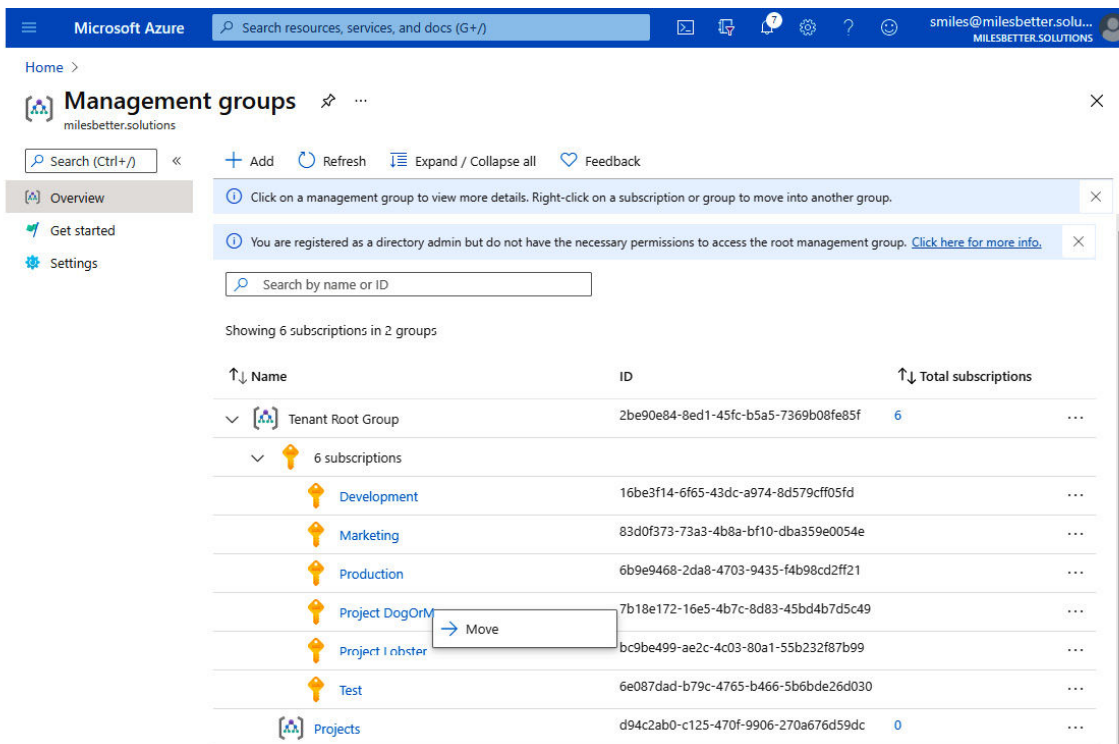


Figure 3.33 – Moving management groups

10. In the **Move** blade, select the management group to move the subscription to and click **Save**; repeat for each subscription to move:

The screenshot shows the Microsoft Azure portal interface. The main blade is titled "Management groups" for the user "milesbetter.solutions". It displays a list of 6 subscriptions under the "Tenant Root Group". The subscriptions are: Development, Marketing, Production, Project DogOrMuffin, Project Lobster, and Test. A "Projects" management group is also visible at the bottom of the list. A "Move" dialog is open on the right, showing the subscription ID "7b18e172-16e5-4b7c-8d83-45bd4b7d5c49". The dialog prompts for a "New parent management group" and shows a search box with "Projects (d94c2ab0-c125-470f-9906-270a676d59dc)" selected. Below the search box, it indicates "policies that are applied." The dialog has "Save" and "Cancel" buttons at the bottom.

Name	ID
Tenant Root Group	2be90e84-8ed1-45fc-b5a
6 subscriptions	
Development	16be3f14-6f65-43dc-a974
Marketing	83d0f373-73a3-4b8a-bf10
Production	6b9e9468-2da8-4703-943
Project DogOrMuffin	7b18e172-16e5-4b7c-8d8
Project Lobster	bc9be499-ae2c-4c03-80a
Test	6e087dad-b79c-4765-b46
Projects	d94c2ab0-c125-470f-990

Figure 3.34 – Selecting management groups

11. The new hierarchy can be seen in the management group's blade; the **Projects** management group now contains two subscriptions:

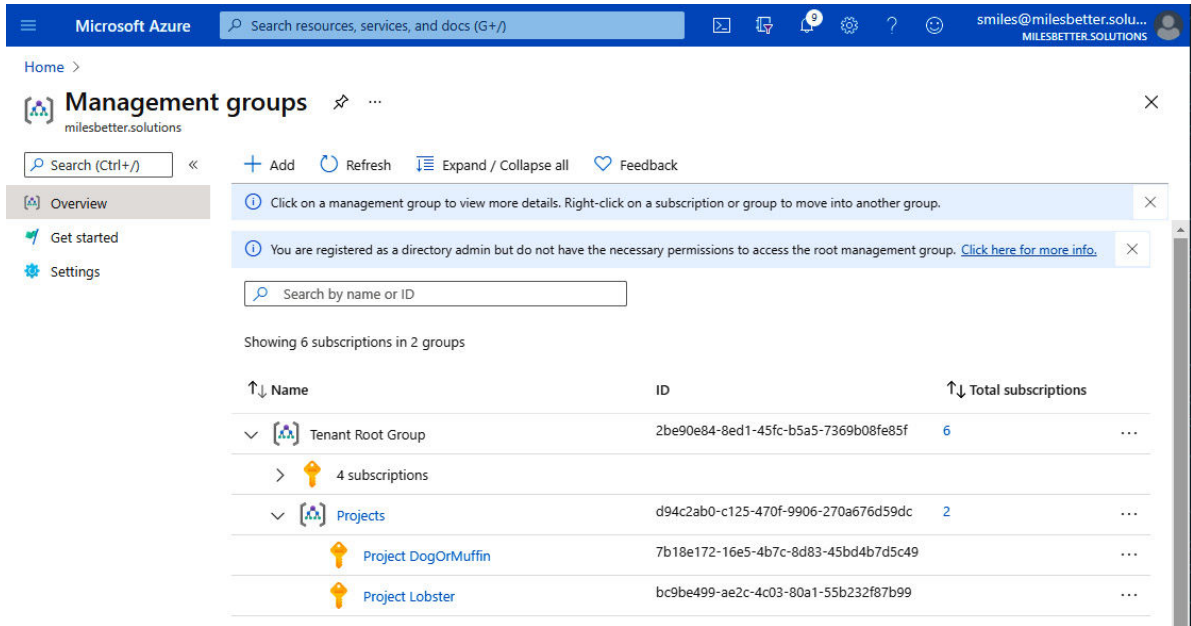


Figure 3.35 – Reviewing the hierarchy

- Clicking on the management group name opens that particular management group. The **Projects** management group now has two subscriptions (called **Project DogOrMuffin** and **Project Lobster**); **Governance** controls can now be targeted at the subscriptions within the management group:

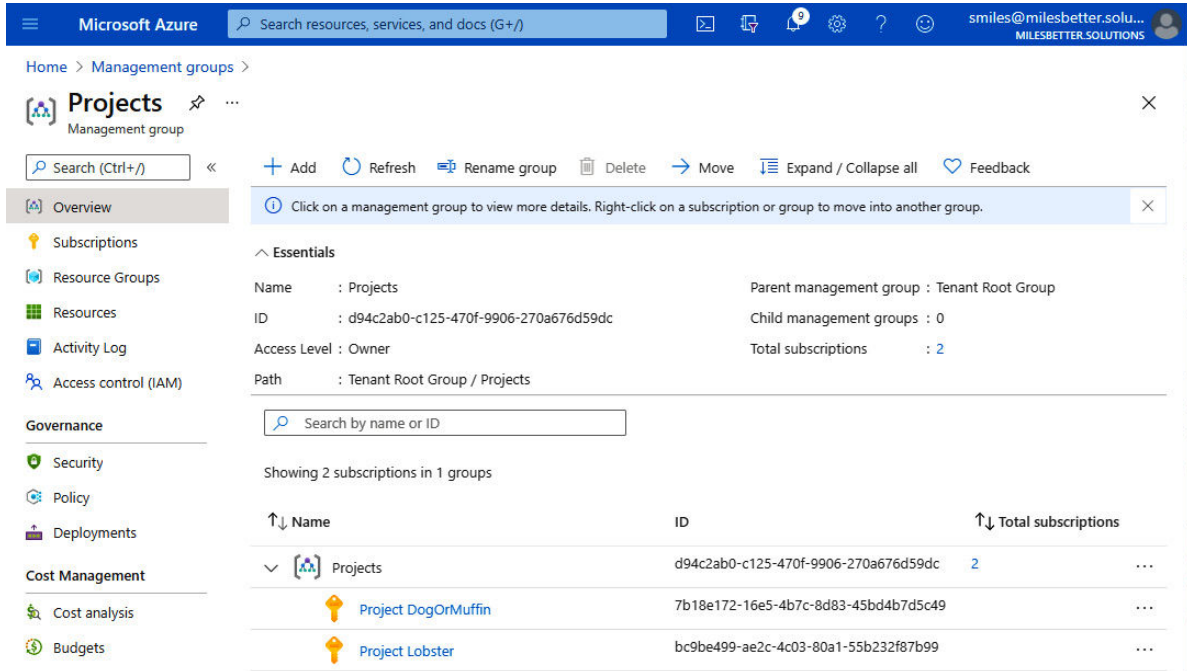


Figure 3.36 – Reviewing the management group

In this exercise, we created a management group and moved two subscriptions into that management group. In the following exercise, we will look at Azure access assignment through RBAC.

Exercise 2 – Azure access assignment

In this section, we will look at the steps to assign access to an Azure subscription. You can follow the same steps to assign access to other resources.

To recap from a previous section, *Azure subscription access control*, where access management was covered, RBAC provides the ability to assign multiple accounts different access levels to a subscription as needed by an organization.

Follow these steps to assign access to a subscription:

1. Log in to the Azure portal at <https://portal.azure.com>. You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the subscription you wish to give access to, click **Access control (IAM)** on the **Subscriptions** blade's vertical menu:

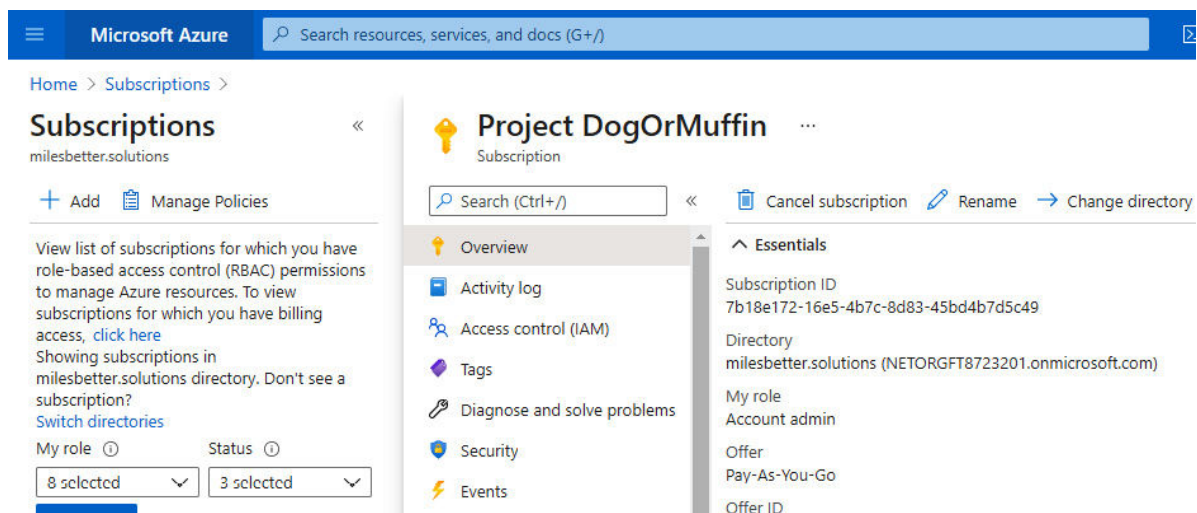


Figure 3.37 – Subscription access

3. To assign a role to another user in the directory, click **+ Add** and select **Add role assignment**:

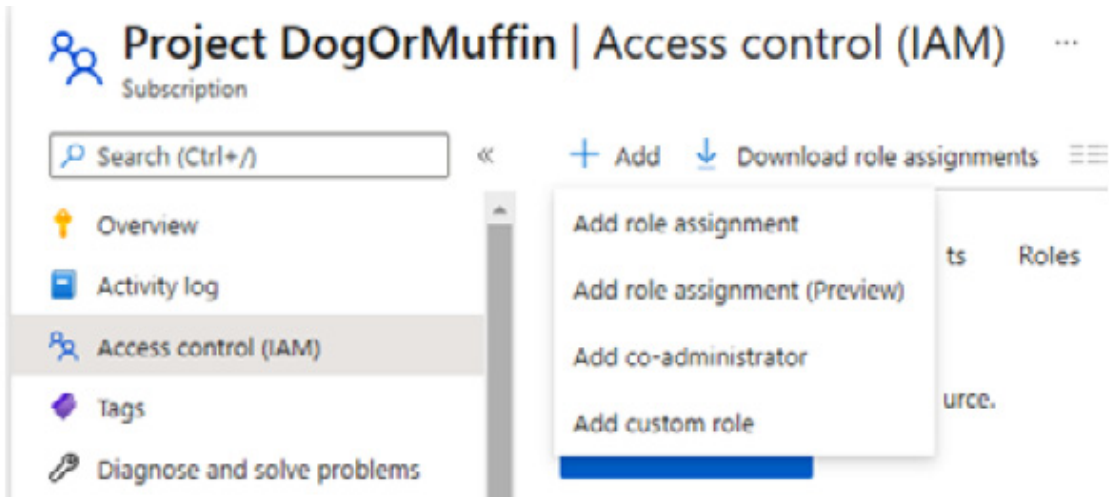


Figure 3.38 – Assigning roles

4. In the **Add role assignment** blade, select the role to assign a user.
5. Then, select the user(s) to assign that role and click **Save**.
6. You will receive a notification that this was completed successfully.
7. Navigate back to the **Role assignments** tab, and the new user role assignment can be seen:

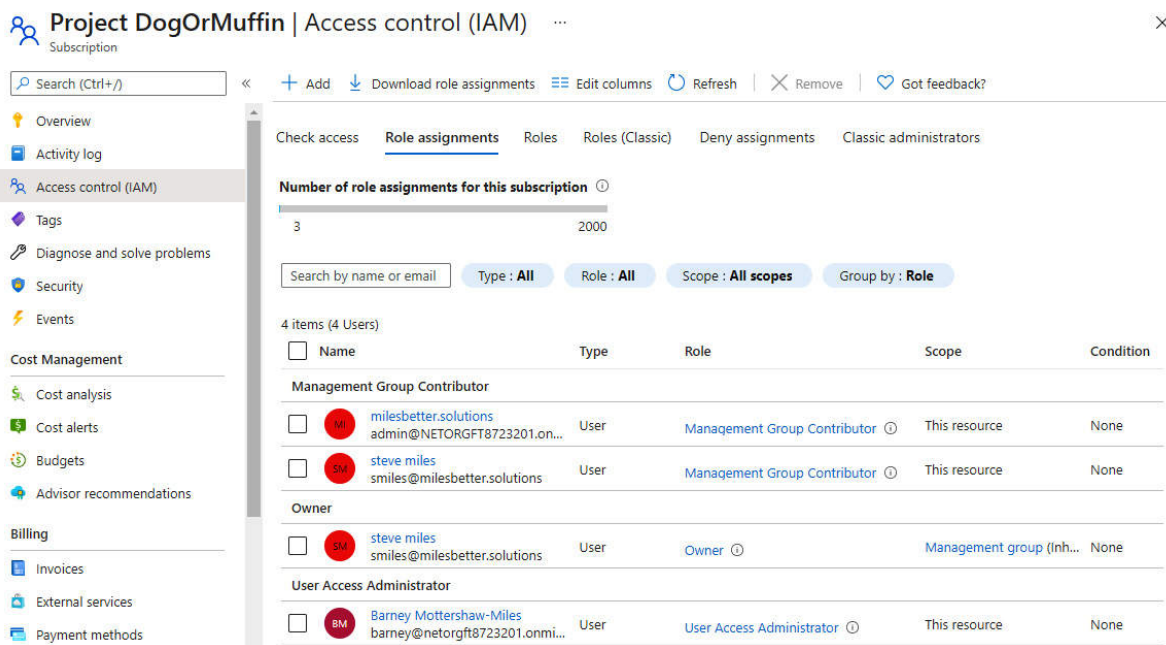


Figure 3.39 – Reviewing assigned access

In this exercise, we added a role assignment to a user through RBAC. In the following exercise, we will look at creating a resource group.

Exercise 3 – resource groups

In this section, we will look at the steps to create resource groups.

To recap from a previous section where resource groups were covered, a resource group is a logical entity that groups resources that share the same life cycle, permissions, and policies.

IMPORTANT NOTE

In contrast to management groups, resource groups are a flat structure. There is no hierarchy or nested resource groups. There is no root/branch or parent/child relationship.

Follow these steps to create a resource group:

1. Log in to the Azure portal at <https://portal.azure.com>. You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type in **resource groups**; click on **Resource groups** from the list of services shown.
3. In the **Resource groups** blade, click on the **Create resource group** button or use the **+ Add** button from the top menu bar of the blade.
4. In the **Create a resource group** wizard blade, the **Basics** tab provides the following information:
 - a) **Subscription**: Select the subscription to create the resource group in.
 - b) **Resource group**: Enter the name for the resource group.
 - c) **Region**: Select the region that is required.
5. Select **Next: Tags** and add any tags as required. Click **Next: Review + create**.
6. On the **Review + create** tab, review your settings; you may go back to the **Basics** or **Tags** tab and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.

You will receive a notification that the resource group was created successfully.

7. In the **Resource groups** blade, the created resource group can be seen:

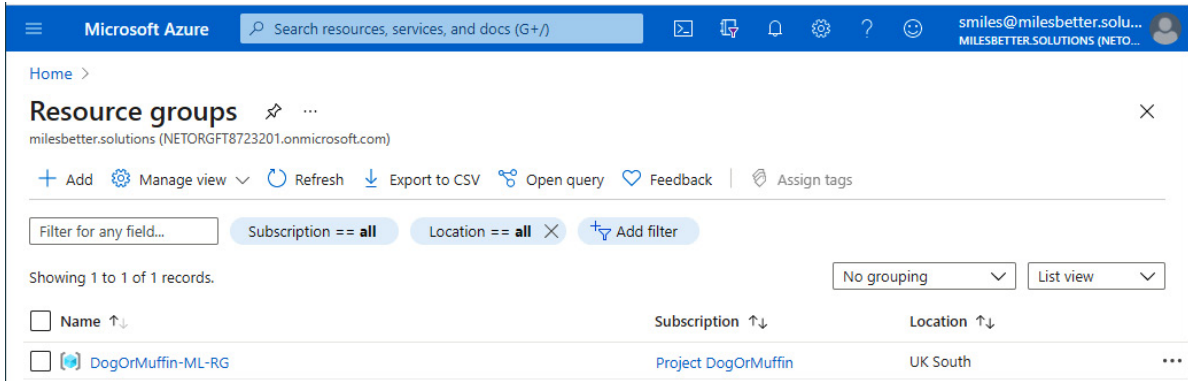


Figure 3.40 – Resource groups

8. Click into the resource group, and resources can now be created within this resource group:

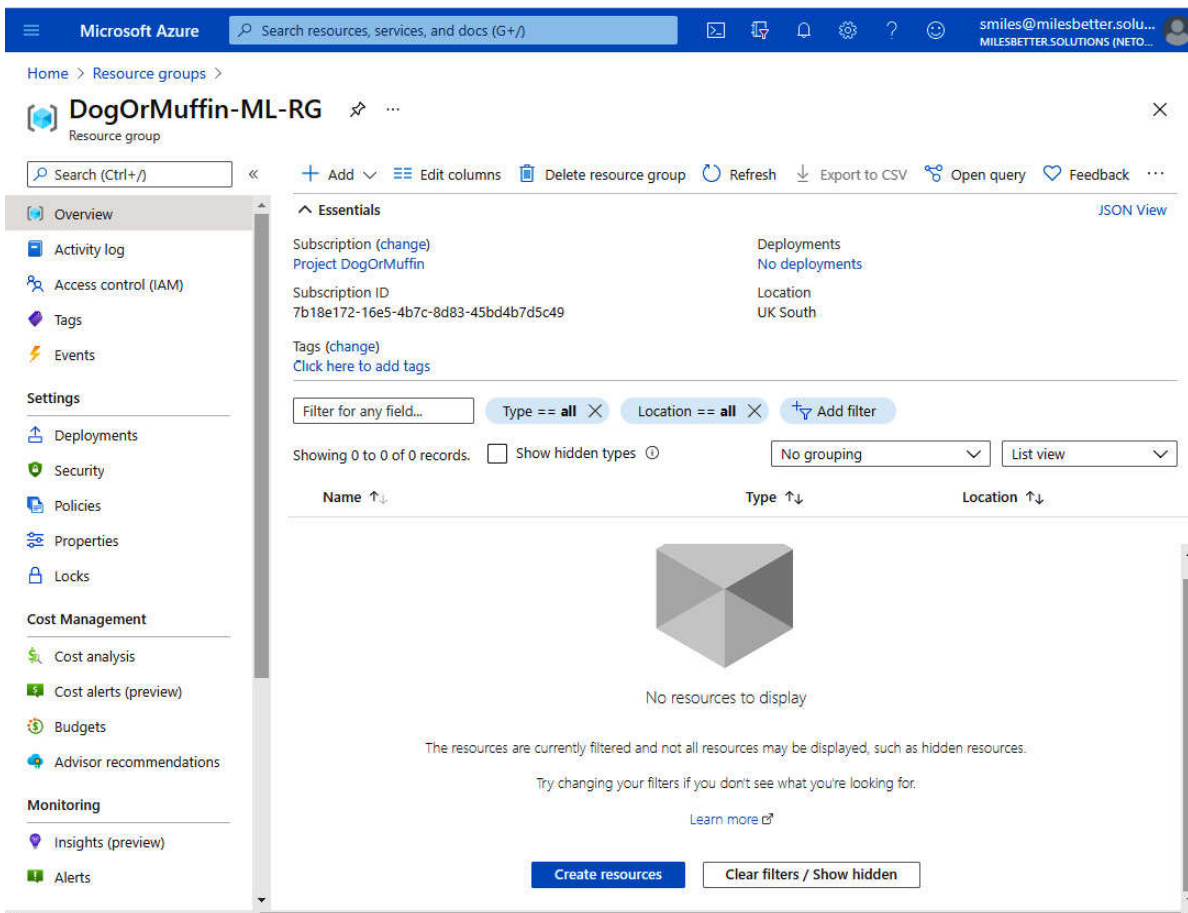


Figure 3.41 – Reviewing the resource group

9. When creating new resources, you can now select the created resource group (created in this exercise as **DogOrMuffin-ML-RG**) to create the new resources within.

In this exercise, we looked at creating resource groups. In the following exercise, we will look at creating proximity placement groups.

Exercise 4 – proximity placement groups

In this section, we will look at the steps to create proximity placement groups.

To recap from a previous section, *Proximity placement groups*, where proximity placement groups were covered, a proximity placement group is a logical entity that groups compute resources, so they are located physically close to each other. This is useful where low latency is a requirement.

Follow these steps to create a proximity placement group:

1. Log in to the Azure portal at <https://portal.azure.com>. You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type in **proximity placement groups**; click on **Proximity placement groups** from the list of services shown.
3. In the **Proximity placement groups** blade, click on the **Create proximity placement group** button or use the **+ Add** button from the top menu bar of the blade.
4. In the **Create Proximity Placement Groups** wizard blade, the **Basics** tab provides the following information:
 - a) **Subscription**: Select the subscription to be billed for this resource.
 - b) **Resource group**: Select an existing resource group or select **Create new**.
 - c) **Region**: Enter the region into which resources should be deployed that will be associated with the proximity placement group.
 - d) **Proximity placement group name**: Enter a name for the proximity placement group.
5. Select **Next: Tags** and add any tags as required. Click **Next: Review + create**.
6. On the **Review + create** tab, review your settings; you may go back to the **Basics** or **Tags** tab and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.

You will receive a notification that the proximity placement group was created successfully.
7. In the **Proximity placement groups** blade, the created group can be seen:

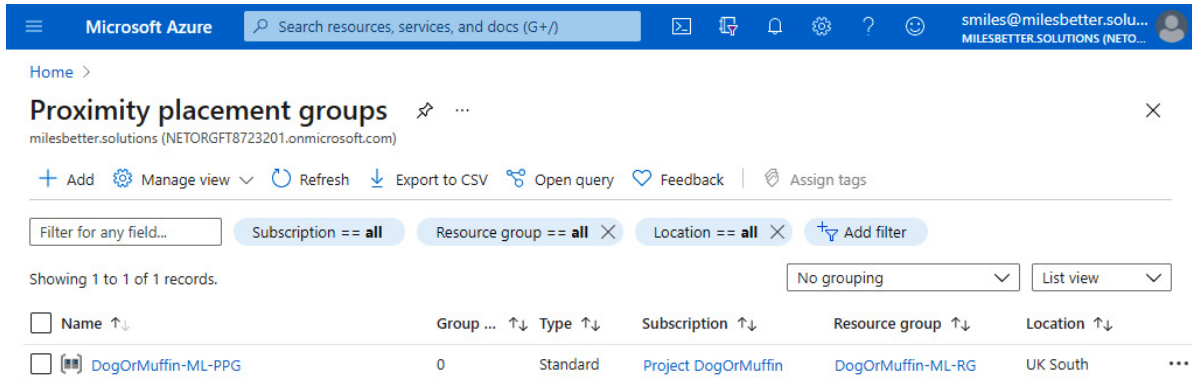


Figure 3.42 – Reviewing the created group

- When creating a resource that can utilize a proximity placement group, select the proximity placement group to use with that resource.

IMPORTANT NOTE

The resource you are creating must be in the same region as the availability set; otherwise, there will be no availability sets available for you to select.

In this exercise, we looked at creating proximity placement groups. In the following exercise, we will look at creating availability sets.

Exercise 5 – availability sets

In this section, we will look at the steps to create availability sets.

To recap from a previous section, *Availability sets*, where they were covered, availability sets are used to provide redundancy for a virtual machine to protect from failures within an individual data center, such as a hardware cluster failure.

Follow these steps to create an availability set:

1. Log in to the Azure portal at <https://portal.azure.com>. You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type in **availability set**; click on **Availability sets** from the list of services shown.
3. In the **Availability sets** blade, click on the **Create availability set** button or use the **+ Add** button from the top menu bar of the blade.
4. In the **Create availability set** wizard blade's **Basics** tab, provide the following information:
 - a) **Subscription**: Select the subscription to be billed for this resource.
 - b) **Resource group**: Select an existing resource group or select **Create new**.
 - c) **Name**: Enter a name for the availability set.
 - d) **Region**: Enter the region into which resources should be deployed that will be associated with the availability set.
 - e) **Fault domains**: Enter the number of fault domains required, or accept the provided value if already at the maximum.
 - f) **Update domains**: Enter the number of fault domains required.
 - g) Click **Next: Advanced**.
5. In the **Advanced** tab, select a proximity placement group if required.
Select **Next: Tags** and add any tags as required. Click **Next: Review + create**.
6. On the **Review + create** tab, review your settings; you may go back to the **Basics**, **Advanced**, or **Tags** tab and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.

You will receive a notification that the availability set was created successfully.

7. In the **Availability sets** blade, the created availability set can be seen:

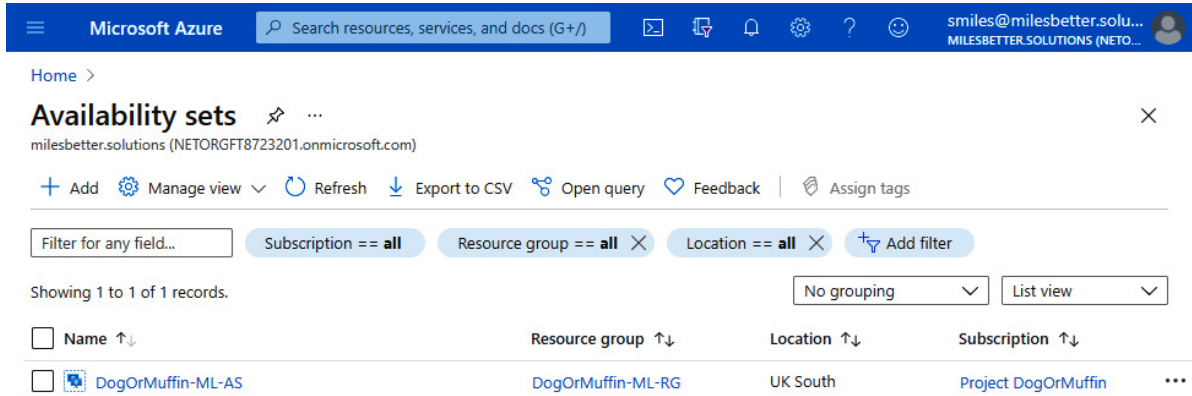


Figure 3.43 – Availability sets

8. When creating a resource that can utilize availability sets, you can now select the availability set to use with that resource.

IMPORTANT NOTE

The resource you are creating must be in the same region as the availability set; otherwise, there will be no availability sets available for you to select.

In this exercise, we looked at creating availability sets.

Summary

This chapter, [Chapter 3](#), *Core Azure Architectural Components*, included complete coverage of the AZ-900 Azure Fundamentals exam skills area *Describe Cloud Concepts*.

We described the *physical* and *logical* core architectural components of the Azure cloud computing platforms. From the *physical perspective*, we looked at the data centers that host the cloud computing resources, the global networks connecting them and connecting users to their resources, the global regions that provide the cloud platform resources, and the availability of these resources.

From the *logical* component perspective, we looked at all aspects of *resource management*. Starting with *Azure subscriptions*, which act as both a mechanism and a boundary for billing and access management, we also covered *management groups*. Next, we covered ARM and *resource groups*, which form the basis for access management and governance, the concepts of **RBAC**, *Azure Policy*, how they differ, and the scenarios in which to use each.

Further knowledge beyond exam content was provided to prepare for a real-world, day-to-day Azure-focused role.

The chapter concluded with a hands-on exercise section bringing together all the skill areas covered in this chapter. The next chapter will outline the core resources available in Azure, such as compute, network, storage, and data, and look at the Azure Marketplace.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900 – Microsoft Azure fundamentals: <https://docs.microsoft.com/en-us/learn/certifications/exams/az-900>
- Exam AZ-900 – skills outline: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- Microsoft Learn: Azure Fundamentals – Describe core Azure architectural components: <https://docs.microsoft.com/en-us/learn/modules/azure-architecture-fundamentals>
- Azure global infrastructure: <https://azure.microsoft.com/en-in/global-infrastructure>
- Azure Edge Zones: <https://docs.microsoft.com/en-us/azure/networking/edge-zones-overview>
- Azure reliability: <https://azure.microsoft.com/en-gb/features/reliability>
- Managing resources: <https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/>

Skills check

Challenge yourself with what you have learned in this chapter:

1. What are the two key components that make up the Azure global infrastructure?
2. What is the difference between an Azure data center and an edge location?
3. What is ExpressRoute?
4. What is the difference between an Azure region and geography?
5. Explain Azure Edge Zones.
6. Explain Sovereign Clouds.
7. What is the Azure availability component that protects against a failure within a data center?
8. What is the Azure availability component that protects against a failure within a region?
9. What is the Azure availability component that protects against a failure across a region?
10. What is the difference between availability sets and Availability Zones?
11. What is a fault domain?
12. What is an update domain?
13. What are proximity placement groups?
14. What does ASR protect against?
15. What are the management scopes?
16. What are Azure management groups?
17. What is an Azure subscription?
18. What is the relationship between subscriptions and Azure AD tenants?
19. How is access to Azure subscriptions granted and controlled?
20. What are the core ARM roles and the differences between them?
21. How is external identity access given to an Azure subscription?
22. What is ARM?
23. What are ARM templates?
24. What are resource groups?
25. How can resource groups be used?

Chapter 4: Core Azure Resources

In [Chapter 3](#), *Core Azure Architectural Components*, you learned how to describe the core architectural physical components available, such as **data centers**, **networks**, **regions**, **availability sets**, **availability zones**, and describe the core architectural, logical components such as **subscriptions**, **management groups**, **resource managers**, and **resource groups**.

This chapter will outline the *core services* available in Azure, including **compute**, **storage**, **networking**, and **databases**, and will also cover the **Azure Marketplace**, which you can use to create these and other services.

This chapter aims to provide complete coverage of the *Describe Core Azure Services AZ-900 Azure Fundamentals Skills Measured* section.

By the end of this chapter, you will be able to do the following:

- Explain **Azure resources** and describe the benefits and usage of Azure Marketplace.
- Describe the benefits and usage of compute services, including **virtual machines (VMs)**, **Azure App Service**, **Azure Container Instances (ACI)**, **Azure Kubernetes Service (AKS)**, **logic apps/functions**, and **Virtual Desktop**.
- Describe the benefits and usage of storage services, including **Blob storage**, **Disk Storage**, **file storage**, and **storage tiers**.
- Describe the benefits and usage of network services, including **Virtual networks (VNETs)**, **VPN Gateway**, **virtual network peering**, and **ExpressRoute**.
- Describe the benefits and usage of database services, including **Cosmos DB**, **Azure SQL Database**, **Azure Database for MySQL**, **Azure Database for PostgreSQL**, and **SQL Managed Instance**.

To support your learning, we will also look at how to create some of the resources that will be covered in this chapter.

The following exercises will cover the creation of key resources:

- Exercise 1 – Azure Virtual Network (VNet)
- Exercise 2 – Storage account
- Exercise 3 – Virtual machine

- Exercise 4 – Azure Container Instance (ACI)
- Exercise 5 – Azure Web App

By the end of this chapter, you will be able to take your knowledge beyond the exam's objectives so that you are prepared for a real-world, day-to-day Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will need the following:

- An Azure subscription so that you can create and delete resources in the subscription. If you do not have an Azure subscription, you can create a free Azure account from this URL:
<https://azure.microsoft.com/free>.
- Access to an internet browser; you will be logging into the Azure portal at
<https://portal.azure.com>.

Alternatively, you can use the Azure desktop app:

<https://portal.azure.com/App/Download>.

Azure resources

In the previous chapter, we covered resource groups and learned that they are a logical grouping of Azure resources; but what is a resource in Azure?

An Azure resource is an Azure entity that can be managed; these resources are the building block components combined to create services, and it is these services we use to provide solutions.

These services and resources can be broken down into more manageable identities and categories, almost like a service catalog of Azure services you can browse through and select to build your solution. There are approximately 200 unique services within the Azure services catalog that can be used to create solutions, which can be made up of several thousand individual resources.

To build a solution, we know we need some compute resources, networking services, storage services, data services, management services, security and identity services, and so on. You can then browse through the categories and view all the options available in each of those that meet your needs for the solution you need to build; you can then take the pre-packaged service and some individual service you need. This service catalog, which allows you to browse all these services and select one to create, is known as the Azure Marketplace; we will look at the Azure Marketplace later in this chapter in the *Azure Marketplace* section.

The following are the Azure resources categories that are available for building cloud solutions at the time of writing:

- **AI + Machine Learning, Analytics, Blockchain**
- **Compute, Containers, Databases**
- **Developer Tools, DevOps**
- **Identity, Integration, Internet of Things, IT, Management Tools**
- **Media, Migration**

As we can see, Azure provides many opportunities to create a cloud solution that can fulfill almost any need or requirement. This section introduced Azure resources

in general and those specific to meet the exam's objectives. In the next section, we will look at the Azure Marketplace and how these resources and additional resources from Microsoft and other providers can be created using the Azure Marketplace.

Azure Marketplace

The **Azure Marketplace** is a searchable service catalog that allows you to find software and services that have been optimized and, more importantly, certified to run on Azure. Not only can you find all the available Microsoft services, but the real benefit and value is that you can find the thousands of solutions and services available from many third-party vendors and other partners where, besides Azure resources, several also offer consulting services for hire as part of the Marketplace offering.

Apart from being able to browse by category to search for a certified firewall appliance to run in Azure, for example, you also have the option to create the actual service directly from the Marketplace; all the services will be deployed using ARM templates. The service creator creates these so that the deployments can be fully automated, and then creates the ARM templates in the background. When executed by you clicking **Create**, a set of parameters will be provided for you to fill in. These will be required for the service to be created in a self-service manner.

The Azure Marketplace has a public-facing website and is also available while you're directly logged into the Azure portal. The URLs for these sites are as follows:

- <https://azuremarketplace.microsoft.com>
- https://portal.azure.com/#blade/Microsoft_Azure_Marketplace/MarketplaceOffersBlade

The portal allows you to browse through the categories of services, such as Compute, Identity, Analytics, and Storage, to find something you may wish to create for your solution; maybe you want to check whether the vendor you want to use has a certified service that can be created in Azure.

Several filters can be used to find the service you are looking for, as shown in the following screenshot:

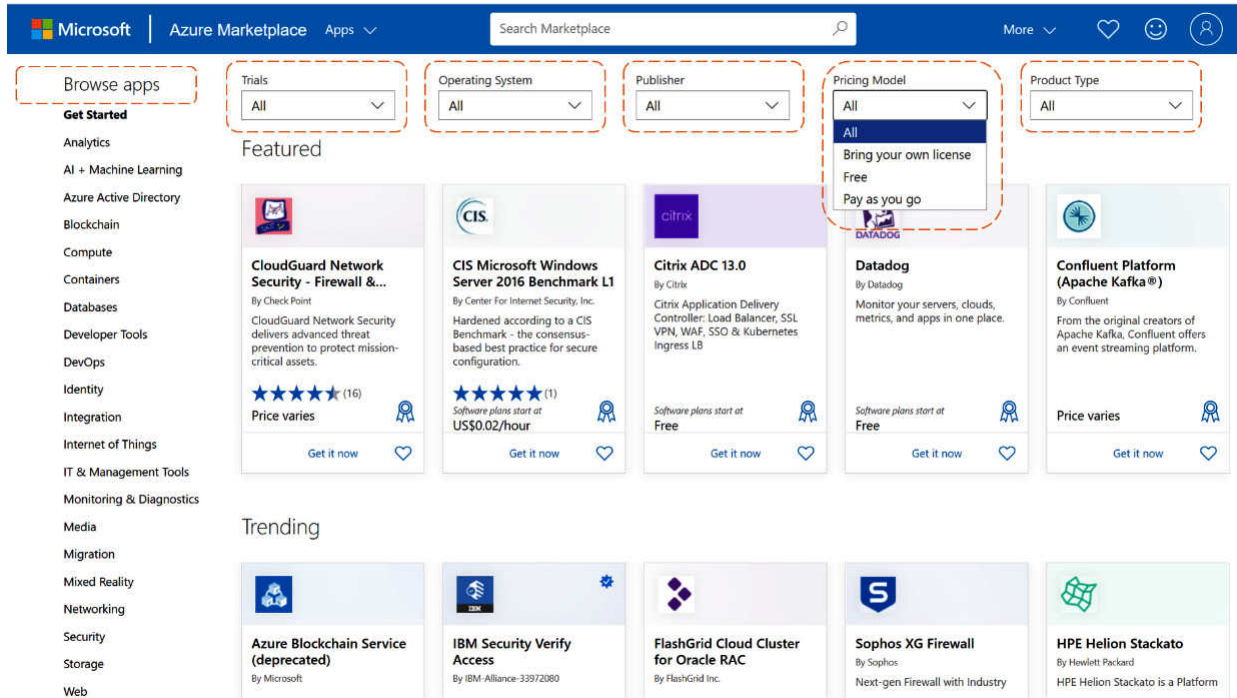


Figure 4.1 – Azure Marketplace

This section introduced the Azure Marketplace. In the next section, we will look at Azure compute services.

Azure compute services

Compute services are one of the core building block services we will be looking at in this chapter. The term compute can be described simply as a platform to execute code on; it's what runs your software and processes your data.

You must understand the following compute services since they are outlined in the exam's *Skills Measured* section; that is, *Describe Core Azure Services*:

- **VM**: A software version of a physical computer
- **Azure App Service**: A Microsoft-managed application hosting service
- **ACI**: A Microsoft-managed container hosting service
- **AKS**: A Microsoft-managed container orchestration service
- **Virtual Desktop**: A Microsoft-managed desktop and app virtualization service

This section introduced the available Azure compute services that can be used to build a solution. In the next section, we will look at the VM compute service.

Virtual machines

This section will introduce **VMs**, a compute service that you must understand, as outlined in the exam's *Skills Measured* section: *Describe Core Azure Services*.

VMs are IaaS compute service resources; they are the common building blocks of any Azure solution. VMs are the virtualization of the physical computer resources of CPU and memory; they are a software emulation (*copy*) of physical hardware computers.

VMs are the most appropriate compute service in the following scenarios:

- There is a need to provide complete control of the **operating system (OS)** and any software installed.
- There are customization requirements, such as customizing the OS, any software/applications, and runtimes. Custom and Azure Marketplace images can also be used; the OS can be Windows or Linux distributions.
- In lift-and-shift migration scenarios where the workload cannot be containerized.
- There is a need to extend on-premises computing capacity, perhaps for development and testing, disaster recovery, and business continuity scenarios. VMs can be connected to an organization's network using a VPN and the internet to route the traffic, or the Microsoft ExpressRoute service, a private network connection that bypasses the internet for better performance (low latency requirements). Alternatively, Azure VMs can be isolated and not connected to on-premises environments.

Creating VMs in your solution means that being an IaaS resource component, there are tasks you are still responsible for under the shared responsibility model we looked at in [Chapter 1, Introduction to Cloud Computing](#). These are the tasks you need to carry out as you would for on-premises compute resources, such as configuring and patching the OS, installing and configuring any software, creating backups, as well as securing the VM with network security controls and managing/securing user account access.

This section introduced VMs as a core Azure IaaS compute service. In the next section, we will look at the differences between physical machines and VMs and the benefits of VMs.

Physical machines versus virtual machines

To understand the benefits of VMs over physical servers, in this section, we will explore the physical hardware operating model.

For each application that an organization wishes to operate, historically, there is typically a one-to-one mapping between the application and a dedicated instance of an OS and physical server, meaning that each app required its own physical server. In this scenario, many physical servers would be required, where an organization will have individual dedicated physical servers to run Exchange, SharePoint, **Active Directory (AD)**, files, print, the web, databases, line-of-business applications, and so on.

The following diagram outlines this traditional typical physical hardware operating model, which has a common shared infrastructure connecting and supporting multiple physical servers:

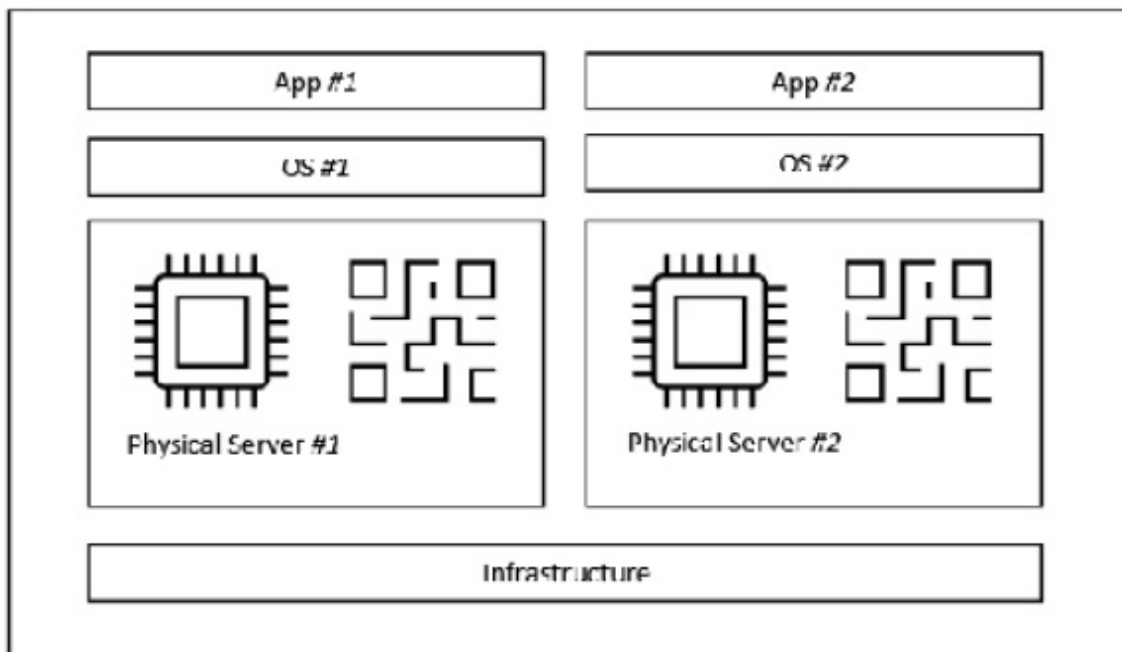


Figure 4.2 – Traditional physical servers approach

This preceding diagram of the typical traditional approach of deploying one physical server per app was (and still is) a very costly and operationally inefficient model. However, virtualization has many benefits to an organization when running on-

premises, and when running in a cloud platform, there can be even greater operational and financial benefits and value, which we looked at in [Chapter 1, Introduction to Cloud Computing](#). This includes no longer having to purchase and maintain hardware in an on-premises facility and having to finance on a CapEx cost model (although maybe leased as OpEx).

When we switch to the virtualization model, we need to understand what virtualization is as a technology concept, its benefits, and what value it provides to both the technology and business personas of an organization.

Virtualization, as a technology concept, is based on abstraction and, specifically, hardware abstraction. The technology layer that's used to achieve this hardware abstraction is known as a *hypervisor*.

Abstraction means removing a dependency and filtering out or ignoring the facts of some characteristics that are no longer relevant for us, which allows us to focus on what is important. So, for example, in the case of virtualization, the VM is no longer dependent on the hardware; if we have abstracted the hardware or removed the hardware from the equation, we no longer need that layer or the detail of it. That is somebody else's aspect to consider – we just take what we need from it.

Likening this to our favorite topic of pizza and our Pizza-as-a-Service from [Chapter 1, Introduction to Cloud Computing](#), we could say that our favorite franchised pizza outlet has abstracted the cooking process in that they have abstracted the oven, as well as the whole kitchen: I requested a specific pizza size that meets my requirements and consumed it; I removed/filtered out (*no longer my concern or care*) the detail of stocking the ingredients and knowing recipes, having a specialty pizza oven, the actual cooking process, and so on.

In the *virtualization versus containerization* section, we will see that *containerization* is all about abstraction at the OS level. *Virtualization* means abstraction at the *hardware* level; beyond this, *serverless* provides abstraction at the *runtime* level.

The following diagram visualizes the approach of virtualization:

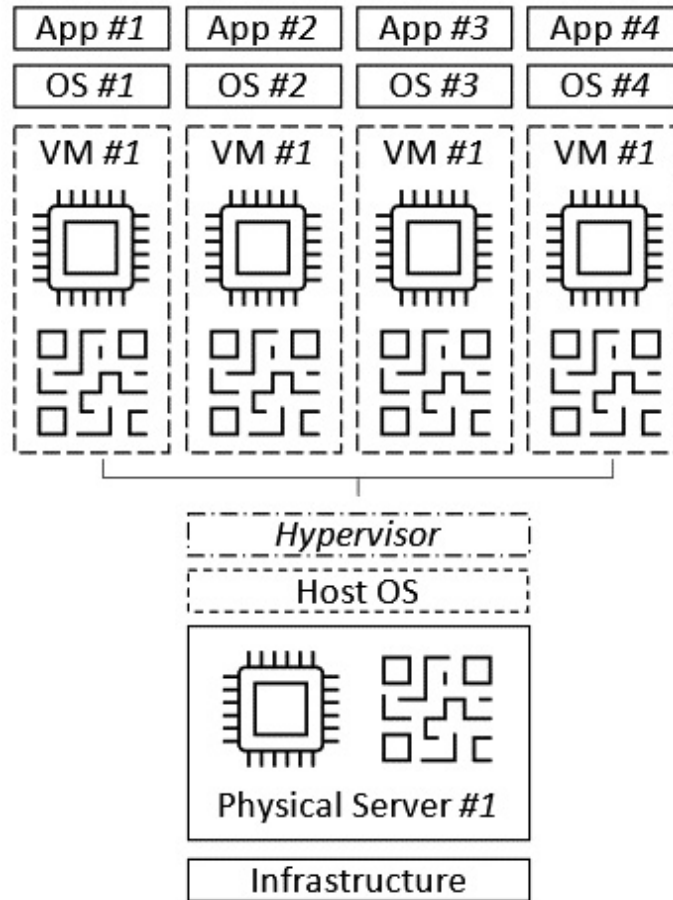


Figure 4.3 – Virtualization approach

This model allows fewer physical servers to run more applications. In the preceding diagram, we have two physical servers to run two apps; with virtualization, we can run four applications from just one physical server in this example.

Each VM shares the underlying hardware resources that the VM is located on, referred to as the host resources. From one physical CPU resource, we can create many virtual CPU and memory resources.

How many VMs you can host from a physical host server will vary on the underlying host's resources ...your mileage may vary.

This section looked at VMs and their advantages over physical servers. In the next section, we will look at the different types of VMs that are available to run your workloads.

VM types

Different workloads have different requirements and need different solutions. Therefore, Azure has many different VM types, and each VM type is tailored to a specific use case and workload type.

VM types are broken down into categories and a family series; the family series identifier is a leading alphabetic character, as seen in the following list. In addition, a naming convention is used to break down the VM types based on their intended use case. Some examples of this include subfamilies, the number of virtual CPUs (this can be expressed as the number of vCPUs), additive features, and versions; we will take a closer look at VM naming conventions later in this section.

Although more of an advanced topic than required for the exam objectives, VMs also support nested virtualization, which allows you to run Hyper-V inside a VM. It is mainly used for testing, training, development, and non-production workloads. Not all VM sizes support nested virtualization, however; as this is liable to change, you should check the following URL for the latest information:

<https://docs.microsoft.com/azure/virtual-machines/sizes>.

Let's take a look at the categories, family series, and intended purpose of various VMs:

- **General Purpose (A, B, D family series):** These VMs have a balanced CPU-to-memory ratio. They are best suited for testing and development (A series only), burstable workloads (B series only), and general-purpose workloads (D series only).
- **Compute Optimized (F family series):** These VMs have a high CPU-to-memory ratio. They are best suited for web servers, application servers, network appliances, batch processes, and any workload where bottlenecks and a lack of resources will typically be CPU over memory.
- **Memory Optimized (E, G, M family series):** These VMs have a high memory-to-CPU ratio. They are best suited for relational databases, in-memory analytics, and any workload where bottlenecks and a lack of resources will typically be memory over CPU.
- **Storage Optimized (L family series):** These VMs have high disk throughput and I/O. They are best suited for data analytics, data warehousing, and any workload where bottlenecks and a lack of resources will typically be disk over memory and CPU.
- **GPU Optimized (N family series):** These VMs have **graphics processing units (GPUs)**. They are best suited for compute-intensive, graphics-/gaming-intensive, visualization, video conferencing/streaming workloads.

- **High Performance (H family series):** These are the most powerful CPU VMs that Azure provides and can provide high-speed throughput network interfaces. They are best suited for compute and network workloads such as SAP Hana.

The preceding family series awareness is all that you need for the exam objectives; however, as this book aims to take your skills *beyond* the exam objectives so that you are prepared for an Azure role, we will take a closer look at the naming conventions. These naming conventions can help you identify the many VMs you will be able to choose from when you create a VM.

For simplicity and readability, we have just kept the core values:

[Family] + [Sub-Family] + [# of vCPUs] + [Additive Features] + [Version]:

- **Family:** The VM family series
- **Sub-Family:** The specialized VM differentiations
- **# of vCPUs:** The number of vCPUs of the VM
- **Additive Features:**
 - a) **a:** AMD-based processor.
 - b) **d:** VM has local temp disk.
 - c) **i:** Isolated size.
 - d) **I:** Low memory size.
 - e) **m:** Memory-intensive size.
 - f) **s:** Premium storage capable; some newer sizes without the attribute of **s** can still support premium storage.
 - g) **t:** Tiny memory size.
- **Version:** The version of the VM family series.

Some example breakdowns are as follows:

- **B2ms:** B series family, two vCPUs, memory-intensive, premium storage capable
- **D4ds v4:** D series family, four vCPUs, local temp disk, premium storage capable, version 4
- **D4as v4:** D series family, four vCPUs, AMD, premium storage capable, version 4
- **E8s v3:** E series family, eight vCPUs, premium storage capable, version 3

- **NV16as v4:** N series family, NVIDIA GRID, 16 vCPUs, AMD, premium storage capable, version 4

This section introduced the different VM types you can use to run your workloads. In the next section, we will look at what else to consider when choosing VMs as the compute service to host your workloads.

VM deployment considerations

Some additional elements to consider when creating VMs for a solution are outlined in this section and visualized in the following diagram:

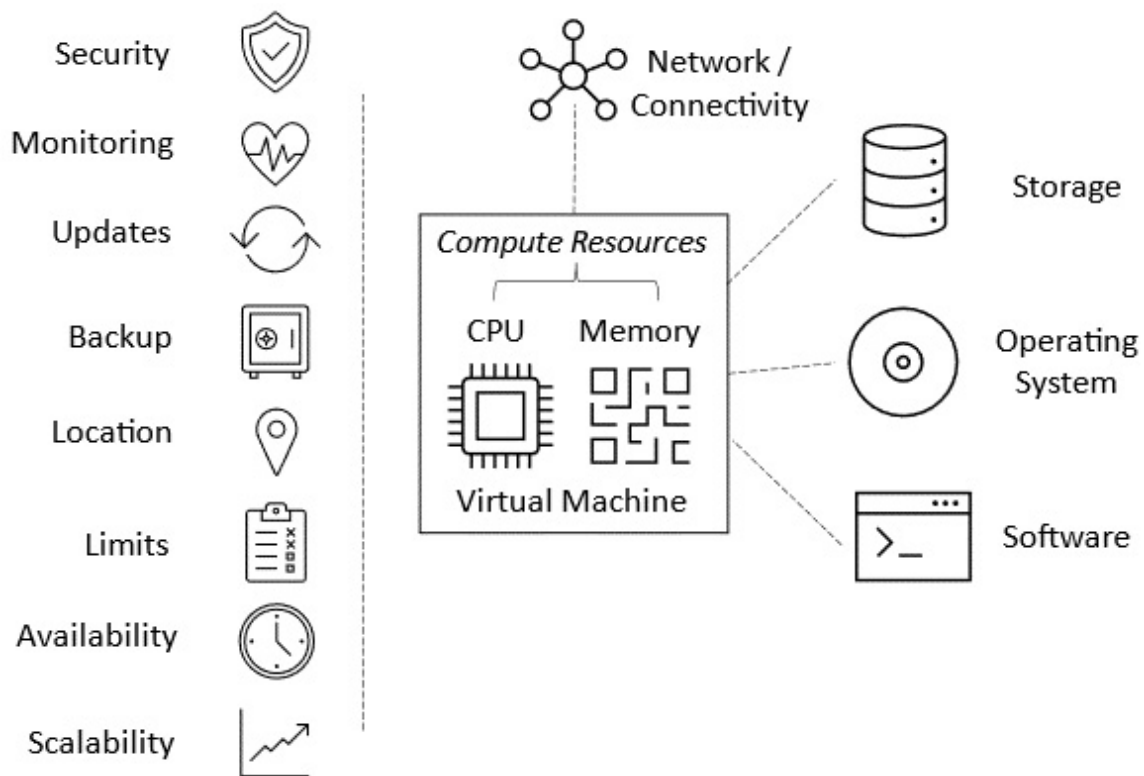


Figure 4.4 – VM components and considerations

Let's look at these additional elements in more detail:

- **Additional VM resources:** A VM includes virtual CPU and memory as its core components; however, you will need to provide an OS, software, storage, networking, connectivity, and security as a minimum for the VM, the same as you would for a physical computer.
- **Location and data residency:** Not all VM types and sizes are available in all regions; you should check that the VM types and sizes required for your solution are available in the region you will be creating resources in. Data residency may also be an important point to consider to

ensure you meet any compliance needs mandated for your organization. Different regions will also have different costs for VM creation.

- **VM quota limits:** Each Azure subscription has default quota limits that could impact the creation of VMs. In addition, there are limits on the number of VMs, the VM's total cores, and the VMs per series. The default limits vary based on the Azure subscription billing type; quota limits for things such as SQL VMs or virtual desktop VMs that require the NV series type can commonly be exceeded, but this can be resolved by requesting a quota limit increase from Microsoft Support. The following link provides further information on these limits:

<https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/azure-subscription-service-limits>.

- **Monitoring:** You can't respond to what you don't know about, so it's vital to have visibility into the performance metrics and operational and security event logs to gain insights. Unfortunately, Microsoft does not automatically monitor VMs or their resources; there are automatically captured activity logs, but this is more of an audit and governance control than performance metrics or operational and security event logs; you must enable monitoring.

You can use Azure Monitor to gain insights into your VM's performance and operational health, as well as using Azure Advisor. In addition, Azure Sentinel, a cloud-native **Security Incident and Event Management (SIEM)** solution, can be used as a single pane of glass and provides a bird's-eye view across all Azure, other clouds, and on-premises resources.

- **Backup:** Microsoft does not automatically back up the OS or software running on your VMs. Under the shared responsibility model, you have complete control of that aspect and it is your responsibility; however, Microsoft's responsibility is to protect the underlying host's hardware and software.
- **Update management:** Microsoft does not automatically update the OS or software running on your VMs. Under the shared responsibility model, you have complete control over that aspect and it is your responsibility; however, Microsoft's responsibility is to update the underlying host's hardware and software. This leads to the next important aspect; that is, availability.
- **Availability:** This is the percentage of time a service is available for use. [Chapter 1, Introduction to Cloud Computing](#), looked at the two core components for addressing SLA requirements for VMs, these were availability sets and Availability Zones, and each provided a SLA. This is important to consider because, as it is Microsoft's hardware and infrastructure that is providing your VM, its resources can and will fail, and this is unavoidable. It is, however, more a case of how you handle the failure when it happens, given the fact that you can't prevent this.

It may be a planned or unplanned update/maintenance event that Microsoft carries out, so it may not be a failure, but it may mean that your VM gets rebooted or moved to another host, which could result in a short service interruption. So, this does not impact you. Microsoft provides measures such as availability sets and Availability Zones so that these events don't impact how your service operates when you implement them.

It is important to note that these measures need to be actioned by you, and by default, they are not configured.

- **Scalability:** This is the ability of a system to handle increased loads while still meeting availability goals. By default, VMs do not support scaling or autoscaling; however, an IaaS resource solution can provide this functionality while still leveraging VMs.

VM scale sets provide this scale functionality. This is an IaaS resource service with built-in autoscaling features for VM-based workloads such as the web and application services, and batch processing. A server farm of identical VMs can be created through VM scale sets, which provides the automatic scaling functionality expected from a cloud model.

This section looked at what to consider when choosing VMs as the compute service to host your workloads. In the next section, we will look at containerization and compare and contrast it to VMs as a compute service.

Container services

This section introduces **container services**, a compute service that you must understand for the *Describe Core Azure Services* exam.

The two container services outlined for the exam are **ACI** and **AKS**:

- ACI is a PaaS service for containers running in Azure.
- AKS is a container hosting platform and orchestration service for managing containers at scale.

Before we look at the ACI and AKS container services further, we will look at the differences between virtualization and containerization.

Virtualization versus containerization

First, let's define these two different compute services approaches:

- **Virtualization** is an approach where the hardware is abstracted; this allows many compute instances (VMs) to share a single host's hardware resources. Each compute instance runs with its own isolated OS.
- **Containerization** is an approach where the OS is abstracted. When we say abstract, what we mean is to remove; that is, remove the requirement to provide that layer. We make the abstracted (removed from thought and consideration) layer the cloud provider's responsibility to provide, keep available, maintain, and so on; we still consume resources from it, but it is a layer that we no longer need to know or care about.

Containers, as a concept, are built around encapsulation and creating standardized software units, packaging an application's code and its dependencies that can be deployed equally seamlessly into a development or production environment with expected, repeatable, and consistent results. The value and benefits of containers are that they are intended to be portable, self-contained computing environments.

A container is a compute unit, similar to a VM; they both have the same goals: *host and execute code*. However, VMs have a lot of overhead, are big, utilize a lot of resources, and have slow boot times. We can consider a container an anti-VM as their characteristics are the complete opposite; they are lightweight, small in size, utilize little resources, and have quicker boot times. You can consider them as

being lighter-weight, more agile, having more efficient compute units than a VM, and being a perfect fit for our digital transformation journey and modernizing our data center and workloads. We may even be brave enough to say that containerization, at some point, will do to virtualization what virtualization did to the process of installing an OS on the physical server bare-metal approach.

This containerization approach allows many compute instances (containers) to share the host software resources of one OS of a single host. Thus, it can be thought of as more of an application delivery model than a virtualization model.

The following diagram visualizes the concepts of virtualization and containerization against the traditional physical approach:

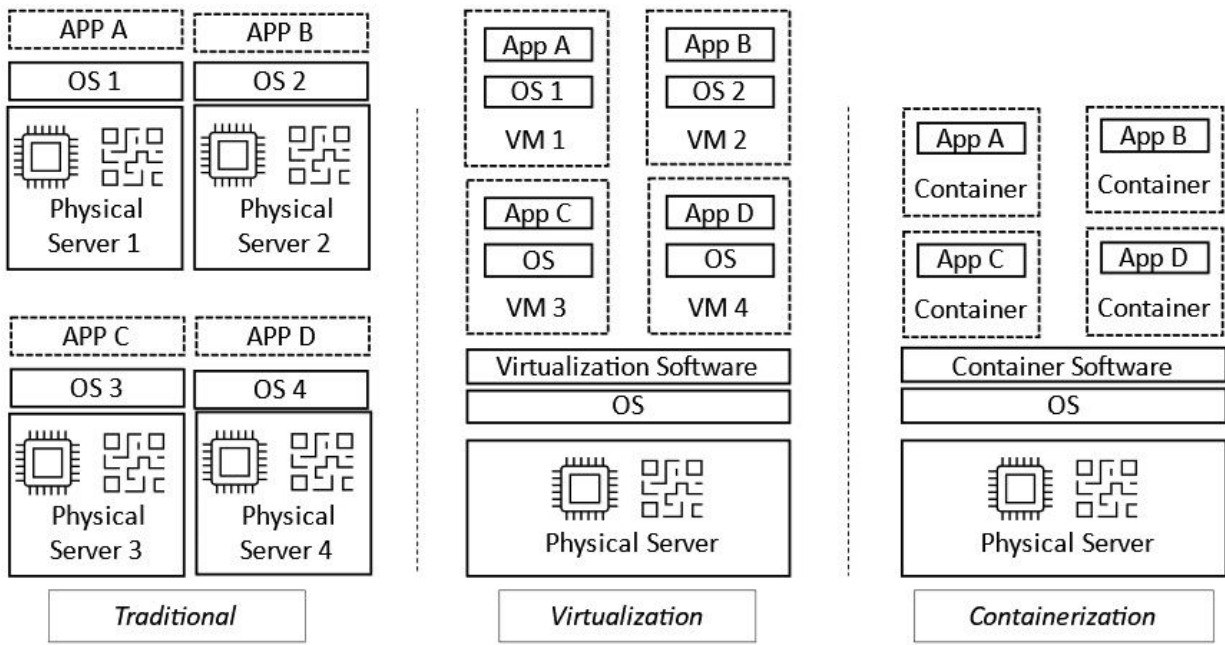


Figure 4.5 – Traditional versus virtualization versus containerization

Let's look at the preceding diagram in more detail, which visualizes the different approaches that can be used to host your application:

- **Traditional approach:** Before virtualization and containerization technologies, applications were installed on an OS that was installed directly on the hardware; you will see this referred to as being installed on bare-metal. This means there was no software (such as a hypervisor) between the OS and the hardware.

The early physical approaches installed many apps on one OS and one physical server (referred to as a piece of tin), but this often resulted in resource starvation, where the resource-hungry app would consume all the resources or software conflicts with processes and libraries (referred to as **dynamic link libraries (DLL)** hell).

This was solved by running each application on its physical server, but this brought other problems as we had over-resourced and under-utilized servers costing a lot of money and doing very little. In addition, it brought physical tin sprawl across the data center, which meant that we needed more physical rack space, which needed more floor space, cabling, power, cooling, and so on. Finally, this meant we needed to focus on data center capacity management, which distracted engineers from the job of deploying and operating apps for the business, which should be their core task.

The bottom line is that the ability of one physical server to deliver one app introduced a scale and inefficiencies problem that was seen as a cost to the business. To IT, this was seen as a cost center, not an innovation or value creation center.

- **Virtualization approach:** Virtualization was an opportunity to change the scenario we saw previously of one app, one physical server; this change was brought about by the ability to abstract (remove dependency from) the hardware.

The benefits virtualizations provided were that they tackled the two key challenges of the traditional physical approach: scale and utilization.

Virtualization meant that we could run multiple VMs and, therefore, multiple applications from one physical server; this gave greater resource utilization and less floor space, power, cooling, and so on than what was required.

The bottom line is that fewer physical servers are required to deliver more apps, translating to fewer costs to deliver the same applications to the business. Due to this, we have a leaner, more agile, and tangible value-delivering IT team.

- **Containerization approach:** Containerization is another evolution in delivering applications to a business; as we learned earlier, this approach changed from abstracting the hardware to abstracting (remove dependency from) the software that brought about the benefits of containerization.

The key benefits and metrics are greater resource utilization/efficiencies/isolation, smaller sizes, faster boot-up, agile app deployment, and no need to provide a guest OS for each app; the app shares the underlying host OS kernel.

The bottom line is that fewer physical servers and OS instances are required to deliver more apps, which, again, means fewer costs to deliver more applications to the business and in a more demand-driven, agile, and standardized manner.

This section introduced containerization, described what containers were and their benefits, and compared containerization with virtualization. In the next section, we will look at ACI.

Azure Container Instances

What is **ACI**? In a nutshell, it's a multitenant, serverless **containers-as-a-service (CaaS)** platform that intends to run single-instance containers in isolation.

What does this mean? Simply put, this means it's a PaaS service for containers running in Azure; you just create a single instance container and start using it as you would a VM. There is no need to manage the VMs to host the containers and no additional services are required to create or manage them.

As containers share the kernel of the host OS it's running on, this means containers are very lightweight, which means faster boot times and that ACI containers can be available in seconds compared to VMs, which have to host their own guest OS and have much slower start times. The time to value is that there's no overhead when creating container platforms, no managing VMs, and no orchestration components. Windows and Linux containers are supported.

It should be noted that there are, of course, VMs underneath – it's that misnomer of serverless again. It just means that you don't need to concern yourself with the details of this; Microsoft provides and manages the underlying hosts, the VMs, the container engine, the orchestration components, and more, leaving you with a compute unit to host and execute your code.

The benefit is that it allows for the most straightforward and quickest entry point to start running your apps in Azure containers, without the need to manage servers or orchestration components.

ACI is best suited to running jobs that are rarely used or scheduled (quarterly, yearly, ad hoc), including batch processing and those that are temporary or demand-driven/event-driven. This also includes jobs that do not interact with other container instances/services or require advanced orchestration, load balancing, or autoscaling, given that these are single-instance containers and are best for single-use and isolated jobs.

This section looked at the ACI compute service. In the next section, we will look at the AKS compute service.

Azure Kubernetes Service

What is **AKS**? In a nutshell, it's a container hosting platform and orchestration service for managing containers at scale. This approach brings the power of the open source Kubernetes platform but delivered as a PaaS-managed platform by Microsoft; think of this as *CaaS*. The service is designed for highly customizable, scalable, and portable workload scenarios.

When comparing *ACI* with *AKS*, it is important to understand the non-functional aspects of cost, complexity, and operations; you can think of each of these solutions at opposing ends of the scale. *ACI* is simple in terms of how it operates and has the lowest cost, whereas *AKS* has the greatest functionality, features, and scale. However, this means it comes with complexity in terms of how it operates and has the highest costs; you must use the *right tool* for the *right job*.

The following diagram visualizes the difference between the *ACI* and *AKS* container services:

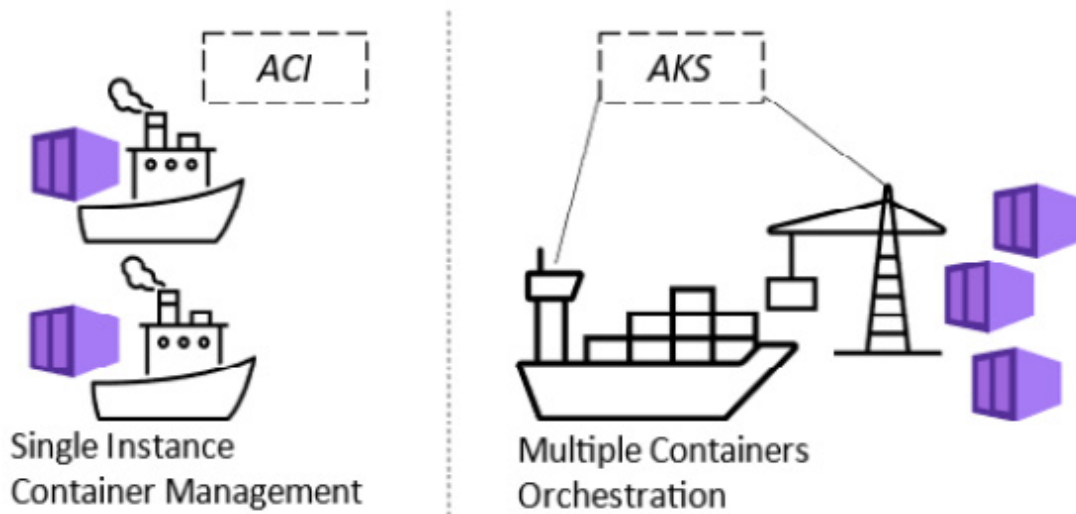


Figure 4.6 – ACI versus AKS

The AKS service architecture has two essential elements: the *master node* is the control plane, which Microsoft manages, while the worker nodes contain *pods*, which the customer manages.

The following diagram shows the high-level AKS service architecture:

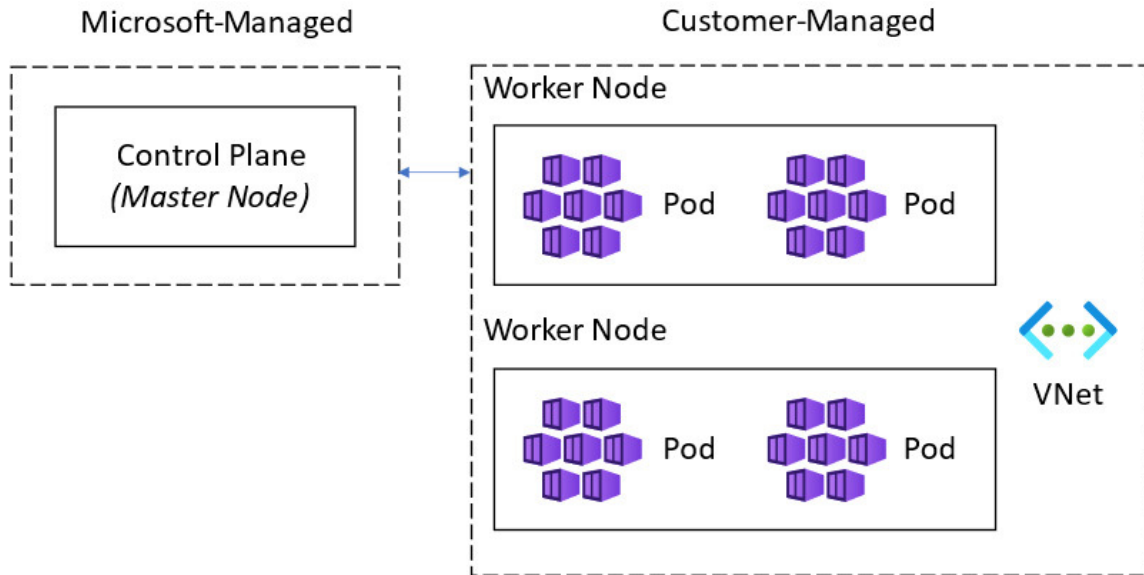


Figure 4.7 – AKS high-level service architecture

The master node is responsible for scheduling the underlying worker nodes; pods and the worker nodes are the VMs that run the node's components and the container runtime.

The following diagram shows the high-level worker node service architecture:

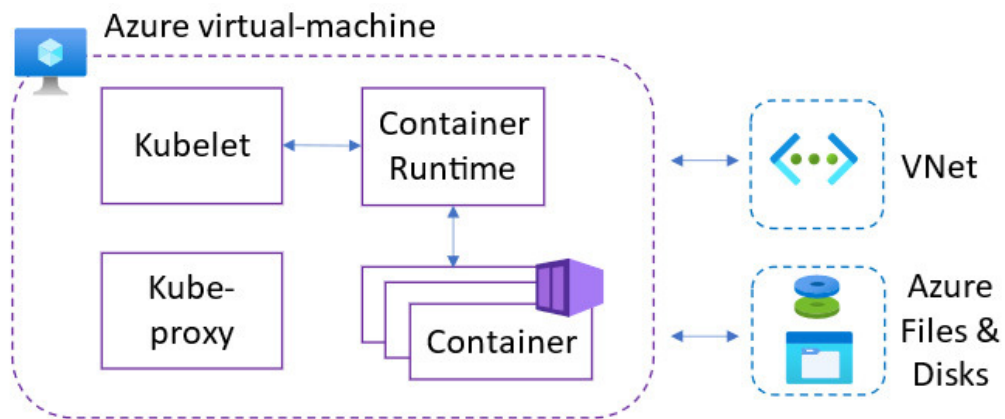


Figure 4.8 – Worker node high-level service architecture

You should size and scale out the VMs in your nodes to support the performance and capacity demands you need. The Azure VM for *cluster nodes* is based on Windows Server 2019 or Ubuntu Linux. Microsoft manages the process of creating

and scaling these VMs; these nodes are billed as regular VMs, which means discounts such as *reservations* can be applied.

This section introduced the AKS compute service. In the next section, we will look at Azure App Service.

Azure App Service

In this section, we will look at Azure App Service, a compute service that you must understand for the *Describe Core Azure Services* exam.

Azure App Service is a PaaS resource and provides a website and code hosting platform that's fully managed by Microsoft. The platform supports Windows and Linux workloads and multiple frameworks and programming languages such as .NET, .NET Core, Node.js, Python, PHP, and more.

The following diagram visualizes how you can host a website or code in an IaaS model versus a PaaS model:

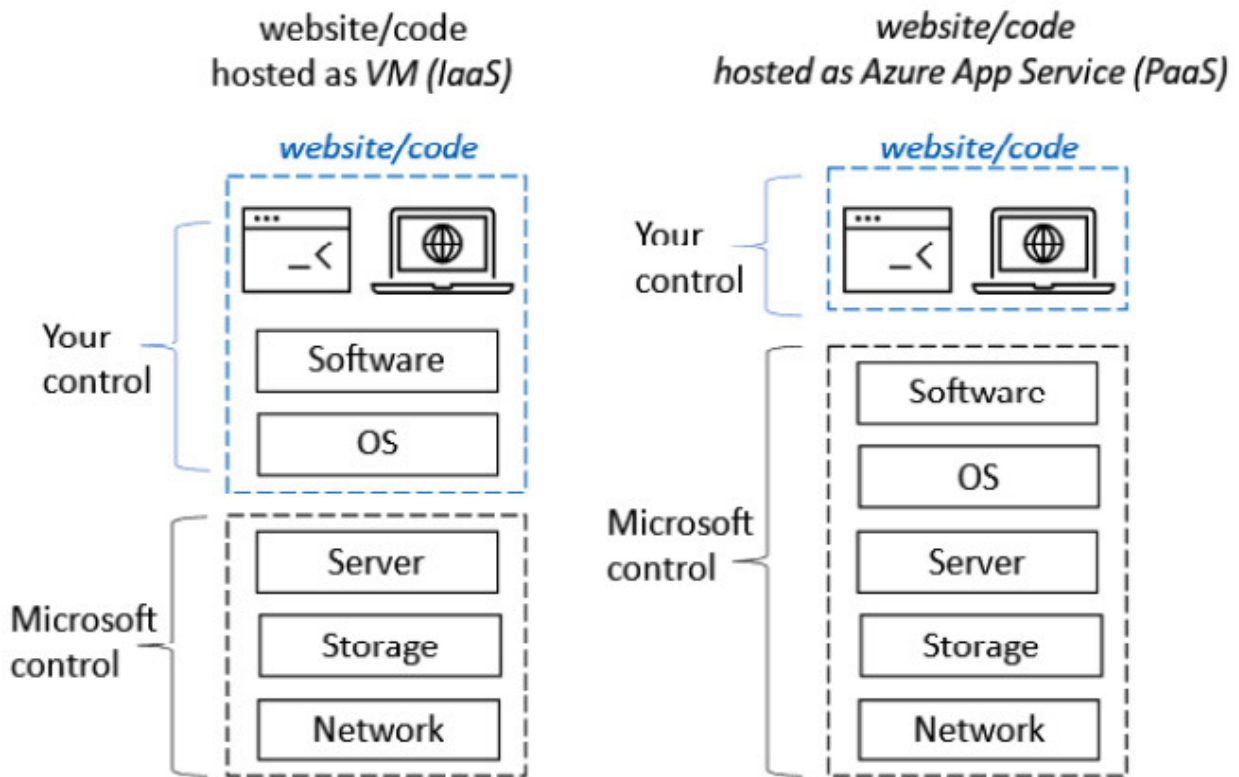


Figure 4.9 – Azure App Service

Being a PaaS service means that this is a hosting platform that's managed by Microsoft; it is responsible for the underlying compute service that will host your

website or code. Therefore, you do not need to manage VMs or containers; you are only responsible for providing your website/code to be hosted.

Each app service runs inside an *App Service plan*. You can think of the App Service plans as a TV subscription service plan or a mobile/broadband service provider plan; each has a tariff with a set of features and functionality you have enabled and can access as a subscriber to that plan. The more fully-featured the plan, the more choice and options you have; you pick the plan that has the most appropriate features that meet your needs.

The following diagram outlines the relationship between all the components of Azure App Service:

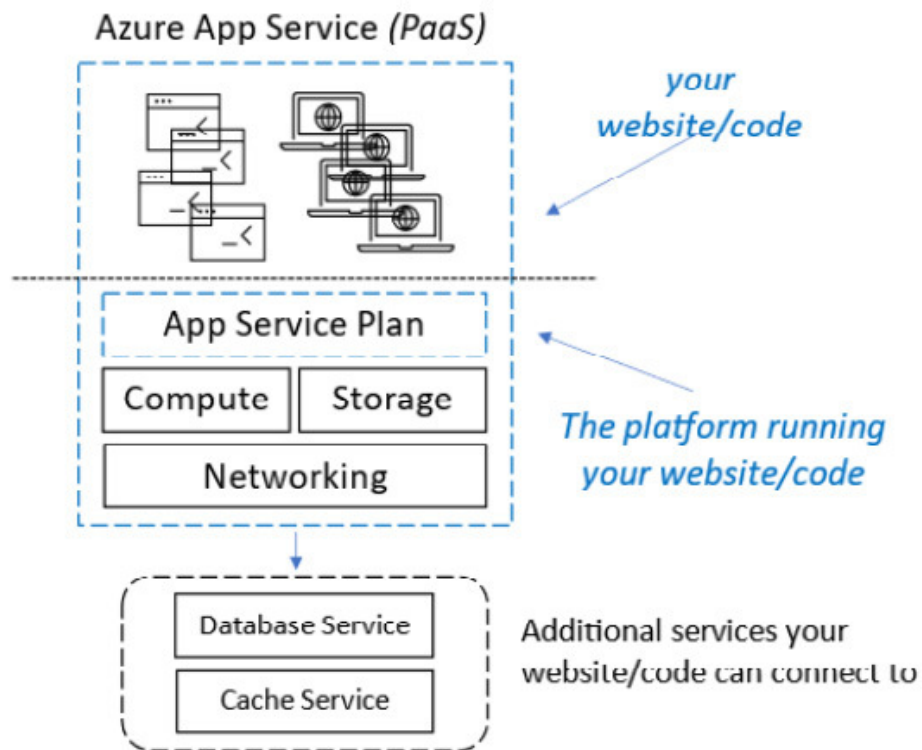


Figure 4.10 – Azure App Service plan

In reality, the App Service plan is closer to that of a server farm; it defines the number of VMs your app service will use, the type and size of VM and its properties, and the region the VMs will be created in. One App Service plan (server farm) may host multiple web apps if enough resources are available from the underlying VM resources.

A web app can only be hosted on a single App Service plan (one set of compute instances, one server farm), but an App Service plan may host many web apps. A web app, when being created, can be added to an existing App Service plan.

Alternatively, you can create a new App Service plan to host your web app/code.

You can select an OS to host your app/code, as well as a runtime stack such as .NET, PHP, or Java, and its version; you can also select the region where your website/code will be hosted. This could have cost or data compliance implications if you must use certain regions based on compliance rules.

App Service has pricing tiers that define the features and functionality available in each tier.

The following price tiers are available:

- **Free and Shared:** Intended for development and testing purposes. No SLA is provided, and your app will run on the same compute service resources as other customers. There is no support for autoscale, hybrid, or VNet connectivity; custom domains are only available in the Shared tier.
- **Basic service plan:** Intended for low-traffic usage that does not require advanced autoscale and traffic management features; it has built-in basic load balancing across instances. Custom domains and hybrid connectivity are supported, but not VNet connectivity.
- **Standard service plan:** Intended for running production workloads. Supports custom domains, autoscale, hybrid, and VNet connectivity.
- **Premium:** Intended for higher scale and performance production workloads. It provides all the features of Standard, plus private endpoints.
- **Isolated:** Intended for mission-critical workloads required to run in a virtual network, in a private, dedicated environment. It provides the same features as Premium.

An App Service plan is billed even when no web apps are running. This is because the VM is created as part of the plan and will remain running until it's deleted, even with no web apps running on the VMs; the only way to stop billing is to delete the App Service plan.

An App Service plan has scale-up and scale-out properties in the portal when configuring an App Service plan.

The scale-up property adds functionality and features; these are part of a pricing tier and you are, in effect, selecting a VM with a pre-configured runtime stack such as .NET, PHP, and Java, and you can select any of the different versions available. The amount of compute instances that can be scaled to is determined by the pricing tier; this is only available for the Basic, Standard, Premium, and Premium V2 pricing tiers. An App Service pricing tier can be changed after creation; this will change the functions and features that are available, as well as the number of compute instances available to run your web app.

The scale-out property adds additional compute instance resources to support running these apps. You can choose to scale manually or set a custom autoscale on a schedule based on any metric available; your pricing tier will determine your scale-out limit; that is, Basic, Standard, Premium, or Premium V2.

This section looked at Azure App Service. In the next section, we will look at the Azure Virtual Desktop service.

The Azure Virtual Desktop service

This section will look at the Virtual Desktop service, a compute service that you must understand for the *Describe Core Azure Services* exam.

Microsoft's Virtual Desktop service is a desktop and application virtualization service that provides access to your desktop and applications from virtually anywhere and on any device.

It is provided as a PaaS service that allows remote users to connect from their devices to their hosted desktops and remote applications in Azure. This access is provided securely and reliably from any location with an internet connection or over a private managed network such as Microsoft's ExpressRoute service; many device types and virtual desktop clients are available to provide the broadest range of connectivity and access methods.

The following are some of the benefits of virtual desktops:

- A full Windows 10 and Office 365 experience.
- Delivers the only multi-session Windows 10 experience.
- A migration path for **Remote Desktop Services (RDS)**.
- Manages Windows 7 End of Support.
- Deploys and scales in minutes.
- Accesses the cloud service anywhere from a web client, or even a desktop client such as Windows, Mac, iOS, Android, and Linux devices.
- Provides centralized identity management and security using Azure AD for role-based access control with Conditional Access and **Microsoft Endpoint Manager (MEM)** and *Microsoft Defender* support.
- Improved remote access security using the reverse connect technology; this means no inbound ports to the session host's VMs, thus reducing the attack surface area.
- Provides the best cost-effective service using pooled desktops through the traditional RDS session host model approach, where many users share one session host VM. Alternatively, they can provide the best performing and secure solution through the traditional VDI approach, where power users or security requirements mandate a level of isolation between user sessions, with

one user per VM (where a desktop cannot be shared between users). This is the personal desktops approach.

- If you do not wish to provide users with access to a full desktop, you can publish the applications that have been installed on the VMs.

The following diagram visualizes the Virtual Desktop approach and its components:

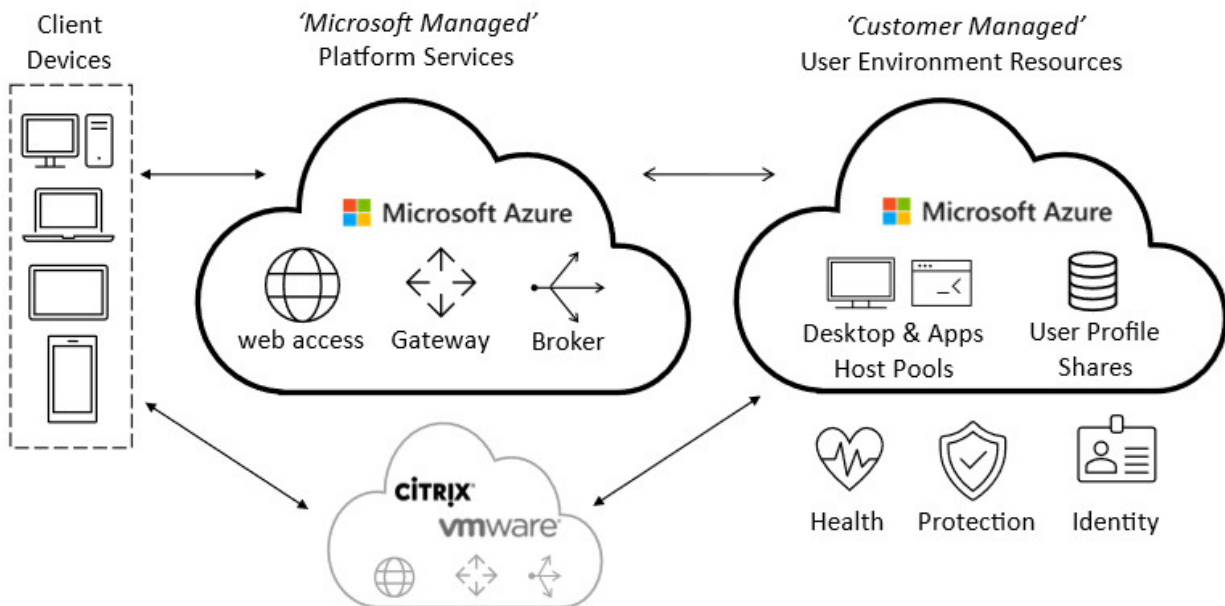


Figure 4.11 – Virtual Desktop

As shown in the preceding diagram, the Virtual Desktop solution is made up of the following elements:

- **Microsoft-provided and managed platform services:** These are PaaS functions where Microsoft provides the managed services of the web access, gateway, and broker roles; these provide the secure connectivity service layer of connecting users with their desktops and published apps.
- **Host pools:** These are collections of VMs. They are the user-assignable entities that will run your users' desktops and publish the remote applications. For example, you may have multiple host pools to meet your needs, and each pool is a collection of VMs that share the same image and configuration that may be assigned to a different set of users; different images and configurations could be made available to different users and teams, based on Azure AD group assignment to different pools.

In addition, there are two load balancing methods: depth mode and breadth mode. *Depth mode* saves costs by fully utilizing a single VM to host users'

sessions before placing a session on the next VM. This is known as *vertical* load balancing. On the other hand, *breadth mode* is the best load balancing method for performance since each user session is created on the next available VM and never on the same VM as the last session that it was connected to (where two or more VMs are available in the host pool). This is known as *horizontal* load balancing.

- **Customer-created desktops:** These are the infrastructure resources that will provide the user's desktops and apps. Host pools form the user's desktop computer layer for the virtual desktop session hosts, which provide the user's desktop and published apps and file shares; file shares will be required to host the user's profile containers.
- **Customer-created remote applications:** Apps no longer need to be installed on the VM OS itself or included as part of the image; through the MSIX app attach functionality, the apps are decoupled from the OS and are then dynamically attached to the VM that a user is connected to for their remote session.
- **User profiles:** Here, the FSLogix profile container technology is used, which allows a user profile to be stored on a **virtual hard disk (VHD)** in the file share location, which is then dynamically attached to the VM that a user is connected to for their remote session, whether it be the full desktop or just the application that has been published.
- **Third party-provided and managed virtual desktop platform services:** Vendors such as Citrix and VMware are part of the ecosystem that provides additional functionalities. You can utilize their presentation and management layers to connect to your virtual desktops and apps.
- **Cost savings:** Cost savings can be made by running the VMs only when users connect to them. Here, you can also consider the use of ephemeral disks to negate the need to persist disks in storage.

In contrast to shutting VMs down to save costs, if the VMs must be running 24/7, then you can save on the pay-as-you-go metered pricing by committing to a 1- or 3-year commitment to reserve the compute capacity, which is called **Reserved Instances (RI)** or *VM reservations*. Note that this only discounts the compute costs of the VM; you will still have to pay the OS costs and for storage, networking, and so on.

Finally, through the **Azure Hybrid Use Benefit (AHUB)**, you can also negate the OS costs for the VMs if you have eligible software licensing; you should

contact Microsoft Support or a licensing specialist partner for guidance in this area.

This section looked at the Virtual Desktop service. In the next section, we will look at the network services available in Azure.

Azure network services

Azure provides a range of **network services** that you can use to communicate with Azure resources.

You must understand the following network services, as outlined in the *Core Azure Services* exam:

- Virtual network
- Virtual network peering
- Virtual private network gateway
- ExpressRoute

Azure network services provides the following capabilities:

- **Allow communication between Azure resources:** Communication paths between Azure resources, such as VMs, are provided through *VNets*; communication between VNets is enabled by using VNet peering (within the same region and across regions). Additionally, service endpoints are used to connect PaaS resources such as storage services and database services.
- **Allow communication with on-premises resources:** You can extend your local on-premises networks into Azure through virtual private networks, which provide encrypted connections over the internet, or ExpressRoute, which allows private, low-latency connections that bypass the internet.
- **Allow communication with third-party services:** You can communicate with other cloud providers, carriers, hosting and data center services providers, and software vendors to, for example, access the APIs or services offered by these third parties.
- **Allow communication with the internet:** Azure can be used as an internet breakout point for Azure resources or for on-premises purposes to access Azure resources. Azure resources such as VMs can accept incoming internet traffic through a public IP address. However, all VMs will typically have private IP addresses for security purposes, and the public IP address should be assigned via a load balancer or other internet-facing gateway resource.
- **Provide isolation and segmentation of network traffic:** VNets are communication boundaries that provide isolation of the address space(s) (also referred to as prefixes) used within that VNet. To allow VNets to communicate, a VPN gateway, **Network Virtual Appliance (NVA)**, or VNet peering must be used. In addition, a VNet's address space can be segmented with subnets; this

allows traffic filtering and routing to be defined at the subnet level. Typically, a VNet will contain one or more private address spaces, although public prefixes are supported.

- **Provide routing of network traffic:** Azure uses a set of default system routes or traffic direction/destination paths (think of this as a set of defined sat-nav traffic routes that define how to get to a destination when coming from a given location); these routes will direct traffic between the subnets of all inter-connected VNets to on-premises networks and the internet; no traffic inspection or filtering is applied by default. You can also apply **user-defined routes (UDRs)**, which we will look at later in this chapter in the *Internal VNet routing* section.
- **Provides filtering of network traffic:** By default, no traffic filtering occurs within a VNet. However, a **Network Security Group (NSG)** can be used, which is a resource that allows you to define inbound and outbound allow and deny rules; you can filter this traffic based on factors such as source IP, destination IP, port, protocol, and more. Alternatively, you can use an NVA, which is a VM that runs network routing and traffic control/filtering software provided by a third party (Barracuda, Fortinet, and WatchGuard, to name a few). This acts as a firewall/packet filter in this scenario.
- **Provides encryption of network traffic:** Through a VPN gateway, traffic that's been sent and received from on-premises or another VNet can be encrypted.
- **Provides name resolution services through Domain Name System (DNS) services:** Azure has an integrated name resolution service for all resources in a VNet. You can configure the VNet with a custom DNS entry for any internal or external DNS server you have; for example, you can set the custom entry to the IP addresses of domain controllers if these handle the DNS for your VMs.
- **Cost implications:** All traffic entering (*ingress*) a VNet within a region is not billed, but all traffic leaving (*egress*) a VNet and region is. This also applies to VNet peering; traffic between these VNets will be billed for egress, not ingress.

This section introduced the core Azure network services and capabilities. In the next section, we will look at Azure VNets.

Azure VNets

An **Azure VNet** is an IaaS resource and enables communication with Azure resources. A VNet represents a software-defined, single-tenant-dedicated and private network in a single Azure region that your resources communicate on; it is dedicated to you and not shared with other tenants in Azure.

Resources in the Azure VNet can access the internet and on-premises resources via public and private networks; users and on-premises resources can also access and communicate with resources in the Azure VNet via these public and private networks. You may have one or more VNets; each can be connected or isolated as required to meet your goals. In addition, several connectivity options support inter-VNet and cross-premises connectivity scenarios.

The following diagram visualizes the VNet and connectivity approach:

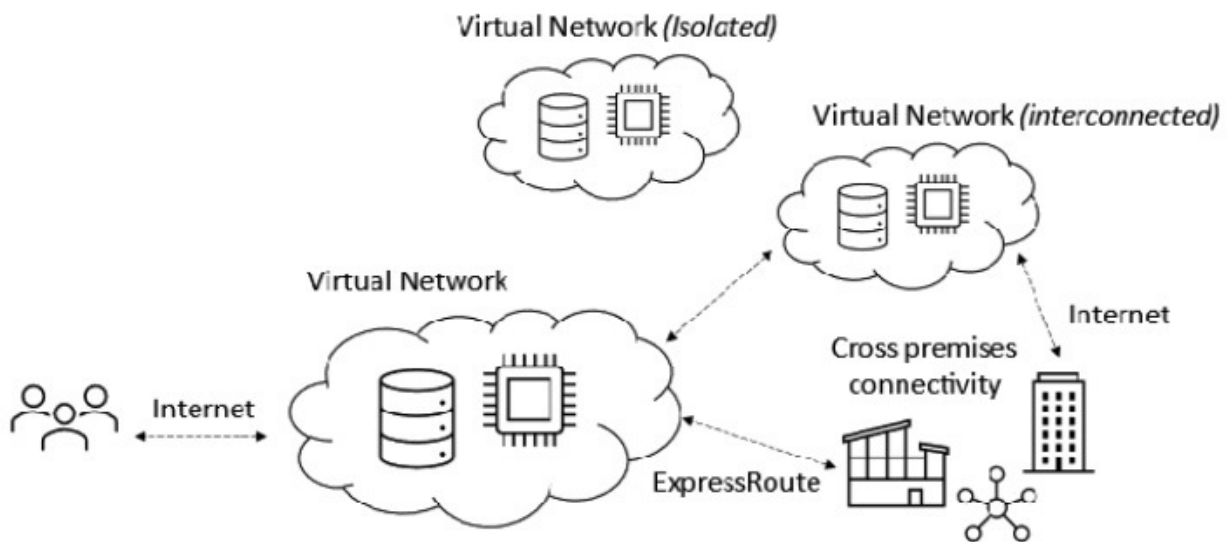


Figure 4.12 – Azure Virtual Networks

The following are some characteristics of VNets:

- VNets act as communication boundaries; by default, Microsoft routes traffic to all the resources in a VNet to allow them to communicate with each other. However, suppose you wish to ensure that the VMs can't communicate, perhaps for compliance reasons. Here, each VM should be placed in a separate VNet.

- If you want the VMs in different VNets to communicate with each other, you must ensure VNet peering or a VPN gateway connects them.
- You can control traffic into a VNet using a VPN gateway (one per VNet), an NSG, or a firewall. A VPN or NSG is a per-VNet control; if you have multiple VNets to control traffic to, then an Azure firewall or third-party NVA is the most efficient option. This has a one-to-many ratio in that one firewall or NVA can control access to multiple VNets across multiple subscriptions. An NSG cannot be applied at the VNet level, only at the subnet level (or the interface level); we will look at Azure firewalls and NSGs in more detail in [Chapter 7, Azure Security](#).
- A VNet belongs to a resource group that belongs to a subscription and can only be part of one region; VNets can span across data center zones but are not available across regions.
- VMs in different subscriptions can communicate with each other, so long as they are part of the same VNet or VNet peering, or a VPN gateway connects the VNets.
- All traffic entering (ingress) a VNet and region is not billed, but all traffic leaving (egress) a VNet and region is billed. This also applies to VNet peering since the traffic between these VNets will be billed for egress, not ingress.
- VNets contain one or more IP address spaces; private IP prefixes are typically used, but public address spaces can also be used.
- VNet address spaces can be segmented into smaller subnets, the same as for on-premises networks. However, note that Azure networking does not allow control at Layer 2 (Data Layer | MAC address) of the **Open Systems Interconnect (OSI)** model; only Layer 3 (Network Layer | IP address) can be controlled. The OSI model concept of how a network functions is beyond the scope of this book and exam objectives, but we are mentioning it here for completeness.
- If you're connecting Azure VNets to on-premises networks, you should ensure that these addresses do not overlap and that they are not duplicated. For example, you must ensure the same matching address space(s) does not exist on-premises. In this scenario, you should create Azure VNets that have unique address spaces that differ from those used on-premises.

This section introduced VNets. In the next section, we will look at VNet segmentation.

VNet segmentation

As we've already learned, VNets, in their simplest form, are a set of communication paths that interconnect systems and other resources and services to those systems. We can think of them as corridors or elevators in a building; we always have entry and exit points into buildings. These are known as doors, but in VNets,

they can be thought of as gateways; there are external doors on the buildings and then internal doors that separate rooms from corridors. We also have different floors, and we may even have buildings connected by interconnecting walkways or skyways in a campus environment.

In VNets, we can create similar constructs to segment a network into smaller sections called subnets; in our building example, a VNet is a building, an address space (also referred to as an address prefix) is a floor (VNets can have multiple address spaces, similar to how a building can have multiple floors), and the subnets are rooms; these are then all connected by corridors, stairs, and lifts, which act as the (human) traffic paths. When you create a VNet, you define an address space; this can be broken down into subnets. Each resource within the VNet will be assigned an IP address.

The following diagram visualizes this concept of VNets and segmentation:

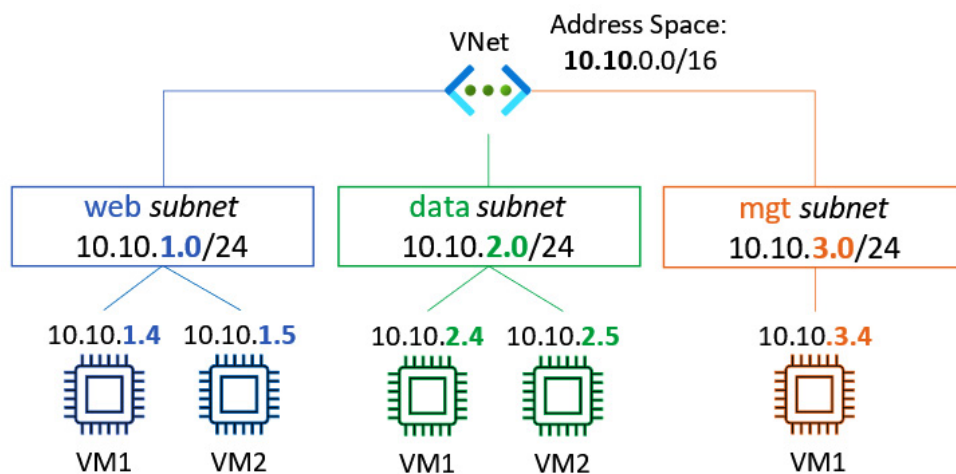


Figure 4.13 – VNet segmentation

The preceding diagram shows an address space that's been created using the **Classless Inter-Domain Routing (CIDR)** format of `10.10.0.0/16`. This has been further segmented into smaller subnets (`/24`), with each resource then assigned an IP address from this range. By default, all traffic can be routed between all the resources in all subnets, routed to other VNets, and routed externally to the internet and on-premises. You can define your routes based on how traffic travels to its destination through UDRs.

It is recommended that you use the following private non-routable (RFC1918) address ranges with Azure VNets:

- 10.0.0.0 to 10.255.255.255 (10/8 prefix)
- 172.16.0.0 to 172.31.255.255 (172.16/12 prefix)
- 192.168.0.0 to 192.168.255.255 (192.168/16 prefix)

Azure reserves five IP addresses in any subnet. When creating a VM, you will notice that the first IP that's assigned in any given subnet will be a .4 address; this is because Azure reserves the first four IP addresses (ranges start at 0, not 1) and the last IP address. Let's look at these reserved IPs in a subnet:

- The first IP address is reserved for the VNet's network address; that is, **x.x.x.0**.
- The second IP address is reserved for the VNet's default gateway; that is, **x.x.x.1**.
- The third and fourth IP addresses are reserved for the VNet's Azure DNS; that is, **x.x.x.2** and **x.x.x.3**, respectively.
- The fifth IP address is reserved for the VNet broadcast address; that is, **x.x.x.255**.

This section looked at VNet segmentation. The following section looks at internal routing.

Internal VNet routing

As we learned earlier in this chapter, VNets have a set of communication paths that interconnect systems. Azure uses a set of default system routes or traffic direction/destination paths that allow resources to communicate with each other.

You cannot edit or remove these routes, but you can create *custom route tables* to control (*redirect*) the direction of traffic (*packets*). This is referred to as *UDR* and allows us to override the default Azure *system routes*. Again, likening this to our sat-nav, this allows us to load in our custom traffic directions to override the manufacturer's predefined loaded ones.

We can apply *UDR* through a *route table* resource within a VNet, combined with an *NVA*. This approach allows network segmentation and traffic control to occur as much as it would in a traditional network.

The following diagram outlines this approach:

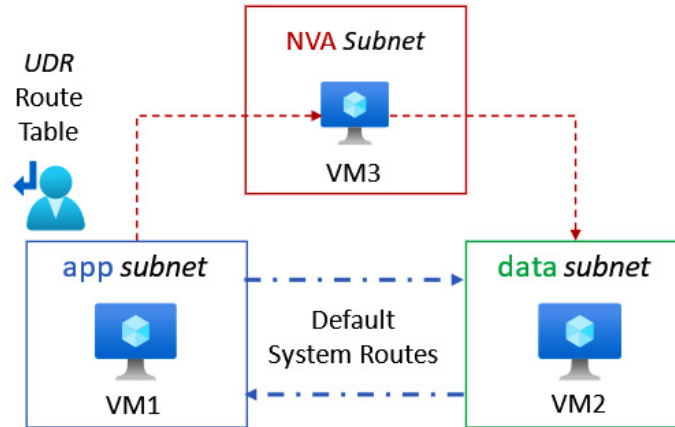


Figure 4.14 – Internal VNet routing

The preceding diagram shows that the default system routing means that there is a communication path between VM1 and VM2; this could mean that if VM1 were to be compromised, it could contact VM2. However, if we apply a route table to the subnet where VM1 is, then any traffic leaving VM1 will now be overridden by the UDR route table and directed to VM3. VM3 will now inspect the traffic and either block or allow the traffic based on the security rules set on the appliance (firewall). In this scenario, if VM1 is compromised, it will no longer communicate directly with VM2 and must be passed to VM3 first.

External VNet routing

External routing is where traffic leaving (egressing) Azure can have its path to its destination determined by cold-potato routing; the alternative is hot-potato routing. Let's take a look at these terms in more detail:

- **Cold-potato routing:** This ensures that traffic remains on the Microsoft global network for as long as possible before it is handed off to a downstream ISP (**Edge Point of Presence (PoP)**) to be delivered to its final destination. This is the most expensive form of egress data from Microsoft regions but has the lowest latency, the best quality, and the fastest performance. This can be likened to a plane courier versus a boat postal service.
- **Hot-potato routing:** This means that traffic leaves (egresses) the Microsoft global network at the earliest opportunity and goes to its destination via the internet, which means passing (*hops*) the packet between points on the internet multiple times. As soon as the packet arrives at a point, it's passed on as soon as possible and not held onto for any length of time; this is where the likening of passing a *hot potato* comes from. This is the cheapest option in terms of data egress costs as it does not stay on the Microsoft network for any length of time; it exits onto the

public network as quickly as possible and is the closest point to the data center. It gets delivered by a cheaper, slower, and less reliable boat postal service.

The following diagram outlines the hot- versus cold-potato routing approach; the cold-potato routed traffic gets delivered to its final destination quicker with low latency and more reliability as it stays on the Microsoft global backbone network longer:

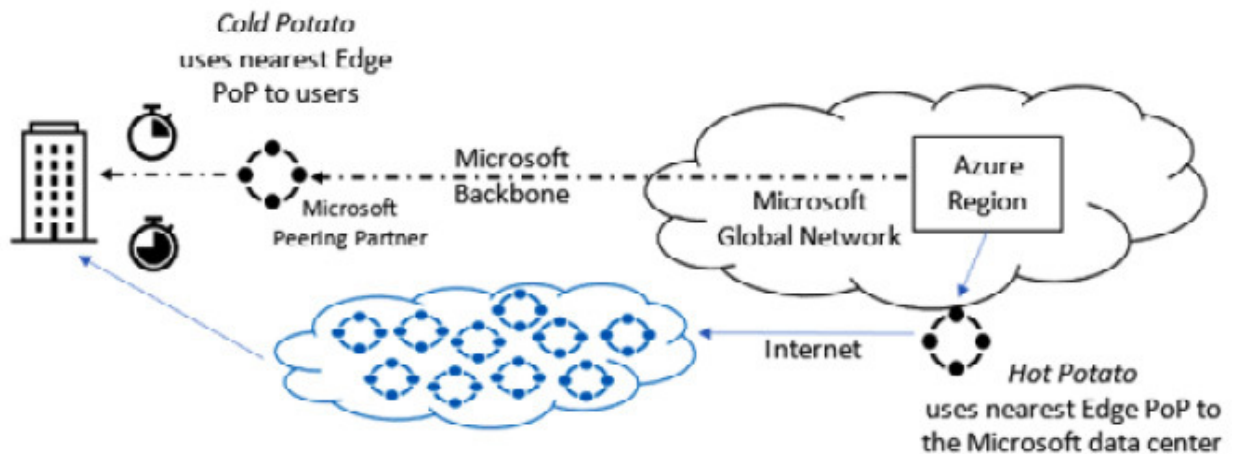


Figure 4.15 – Hot- versus cold-potato routing

The following diagram likens this to delivering a packet to its final destination by using the perspective of air travel versus shipping:

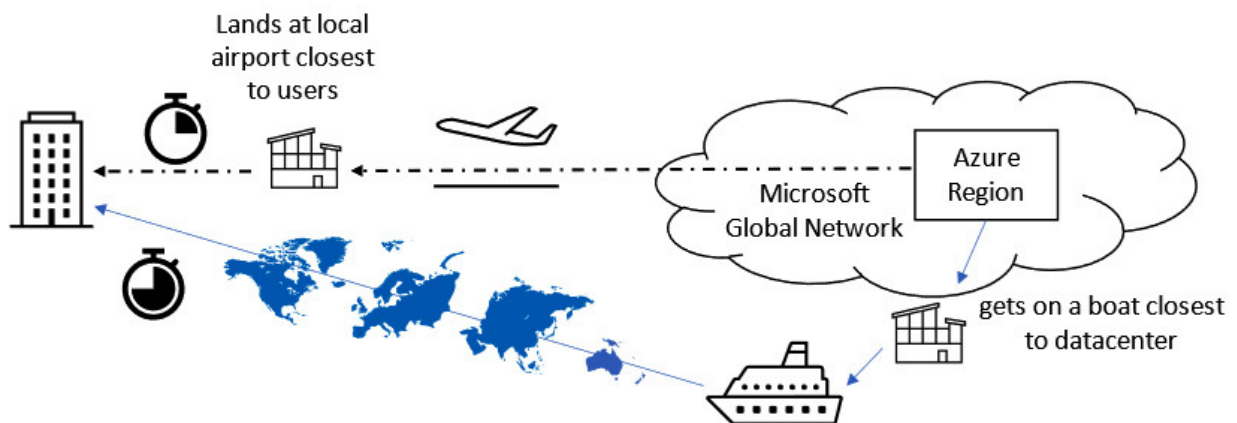


Figure 4.16 – Air travel versus shipping routes

This section looked at VNet external routing. In the next section, we will look at VNet peering.

Virtual network peering

VNet peering allows two or more VNets to be connected seamlessly so that from a communication point of view, the traffic flow appears as though it's on the same network.

Unlike a VPN gateway, whose traffic must route over the internet, traffic between the VNets when using VNet peering is routed over the Microsoft backbone, meaning fast, reliable, low-latency, and secure connectivity between your resources in different VNets. This is the same as traffic being routed between resources within the same VNet. The following diagram visualizes this inter-VNet connectivity:

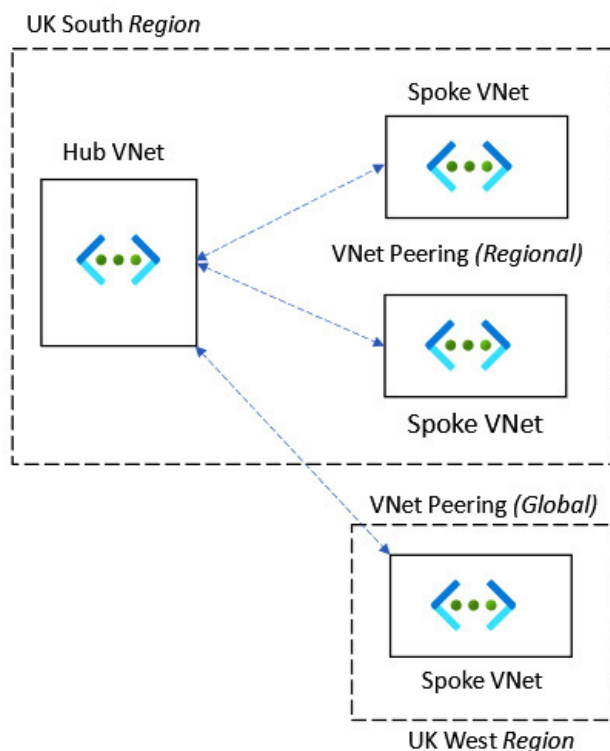


Figure 4.17 – VNet peering

VNet peering also supports *regional* and *global* VNet peering, as shown in the preceding diagram:

- **Regional VNet peering:** This is used to connect VNets from the same region.
- **Global VNet peering:** This is used to connect VNets from different regions.

As seen in the preceding diagram, a hub and spoke design is the most common deployment topology. This section looked at VNet peering. In the next section, we will look at the VPN gateway service.

Virtual private network gateways

A **virtual private network (VPN)** gateway is a service that allows encrypted traffic to be sent through private and secure connections between resources in an on-premises location and an Azure virtual network by using the internet. This allows users and resources to seamlessly connect as though they were connected to the same local network.

A VPN gateway requires several resources to be deployed, as shown here:

- **VNet:** A VNet can only have one VPN gateway; the VPN gateway is a resource associated with only one VNet. When connecting Azure VNets to on-premises networks, you should ensure that these addresses do not overlap, that they are not duplicated, and that the same matching address space does not exist on-premises. In this scenario, you should create Azure VNets that have a unique address space that differs from the one that's used on-premises.
- **Gateway subnet:** A dedicated subnet named GatewaySubnet is created within the VNet; the VNet address space must have space available to allocate this. A /27 (/26 ...) or larger address is recommended since this ensures that enough IP addresses are available.
- **Public IP address:** This resource is the public IP address that's used to connect to the on-premises VPN device (or another VNet's gateway).
- **Local network gateway:** This resource is used to represent your on-premises network location; it defines the VPN appliance that connections will be made from, as well as the network address spaces (also referred to as address prefixes) that you will be connecting from to establish a connection to the Azure resources. A local network gateway can specify a single on-premises network address space or multiple network address spaces.

The local network gateway is configured with an IP address or the **Fully Qualified Domain Name (FQDN)** of the on-premises VPN appliance you will make a connection to; you also specify the IP ranges from the local on-premises network location that you will connect from.

You can modify your local network gateway to reflect any changes in your on-premises network. For example, you can add or remove address spaces (prefixes) if you added a new on-premises network and now need to connect to Azure resources over the VPN associated with this local gateway, and then add these new IP spaces to allow this communication.

If you need to prevent a network from communicating, you can remove these on-premises network spaces from the local gateway, and they will no longer be able to access resources in Azure over the VPN.

- **Connection:** This resource creates a logical connection between the local network gateway and the VPN gateway.

The following diagram shows these resources and their relationships:

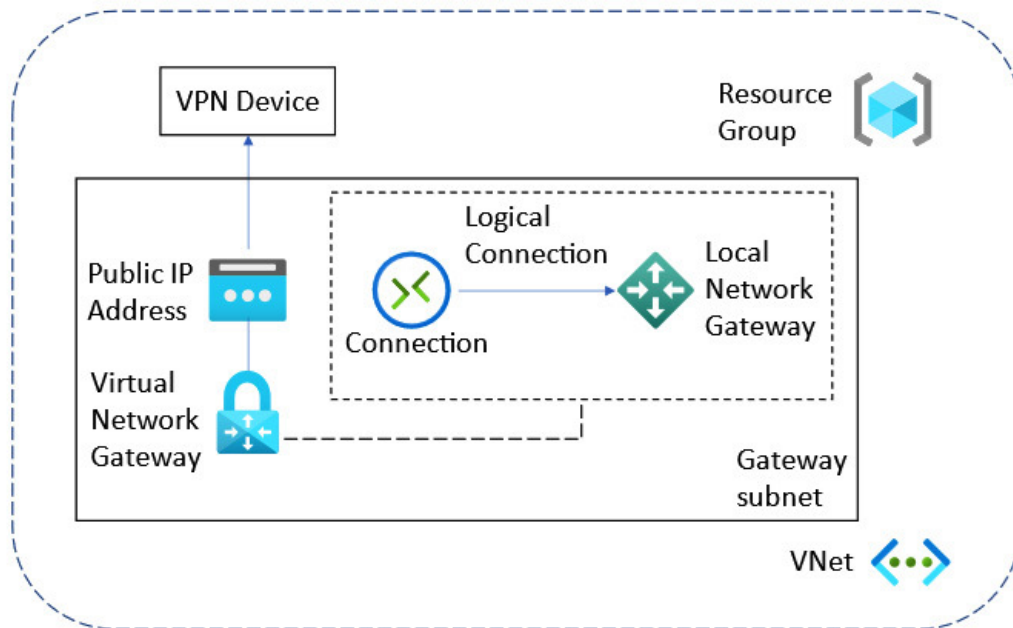


Figure 4.18 – VPN gateway resources

A VPN gateway can support the *Point-to-Site VPN* and *Site-to-Site VPN* connectivity methods. In the next few sections, we will look at these two connectivity methods in more detail.

This section introduced the VPN gateway service. In the next section, we will look at the Point-to-Site VPN connectivity method.

Point-to-Site VPN

This allows you to make a secure encrypted connection to Azure resources by using a single user from a client device. This is intended for remote users who are not connected to the corporate network (and cannot use a Site-to-Site VPN), such as home workers or those traveling. The OpenVPN, SSTP, and IKEv2 protocols are used here:

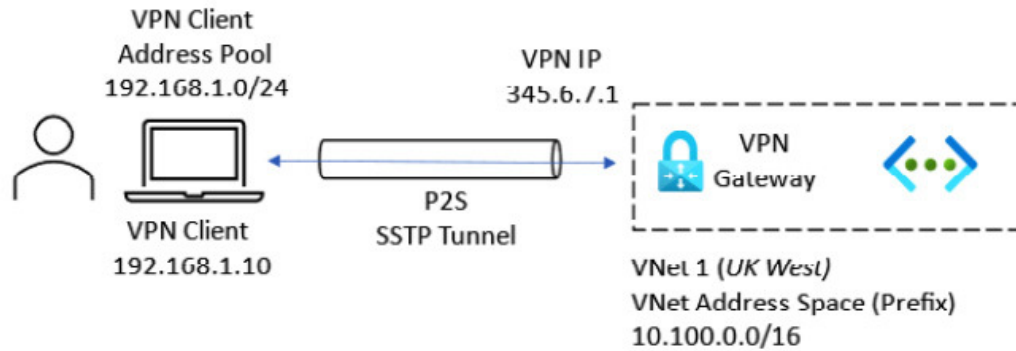


Figure 4.19 – Point-to-Site VPN

In this section, we looked at the Point-to-Site VPN connectivity method. In the next section, we will look at the Site-to-Site VPN connectivity method.

Site-to-Site VPN

This allows you to make a secure encrypted connection to Azure resources in a cross-premises scenario to support a hybrid connectivity solution. The Azure VPN gateway service provides both policy-based VPNs, also referred to as static routing, and route-based VPNs, also referred to as dynamic routing.

Policy-based VPNs may be considered for more legacy scenarios since this is the older approach and is generally used to connect on-premises devices that only support policy-based (static) connections. This is typical since firewalls provide more VPN functionality compared to a fully-fledged and dedicated VPN device. Due to this, route-based VPNs (dynamic) are the recommended approach.

Both support the **Internet Protocol Security (IPSec) protocol** known as **Internet Key Exchange (IKE)**, both versions 1 and 2. The only authentication method that's supported is *pre-shared keys*.

The following diagram shows how to connect an on-premises network to Azure using a Site-to-Site VPN; this topology requires a VPN device to be located at the on-premises site and it must have a public IP address:

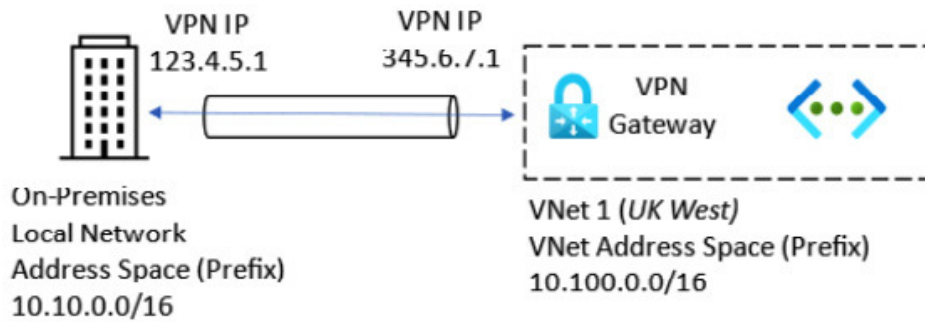


Figure 4.20 – Site-to-Site VPN

The following diagram shows how to connect two Azure VNets using a Site-to-Site VPN:

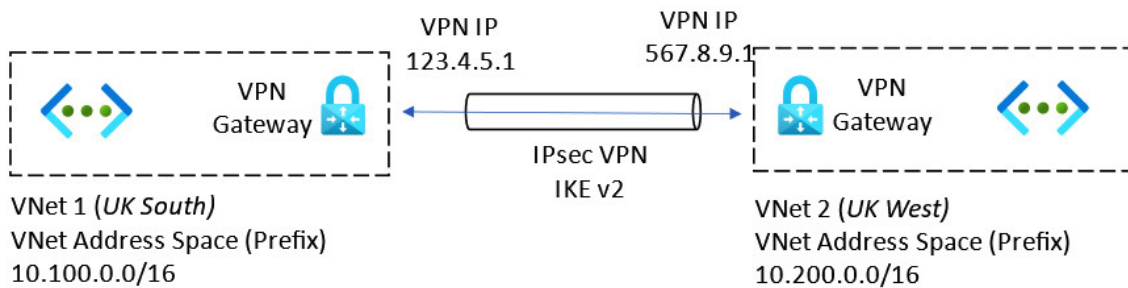


Figure 4.21 – VNet-to-VNet VPN

The alternative to using a Site-to-Site VPN to connect two Azure VNets is to use *VNet peering*; however, this does not encrypt the traffic between the two VNets and uses private addresses. This is a private connection that stays on the Microsoft backbone network, so it may not be a suitable alternative requirement.

This section looked at the Site-to-Site VPNs. In the next section, we will look at the ExpressRoute service.

ExpressRoute

An **ExpressRoute circuit** is a true extension of your on-premises networks into Microsoft's data centers. It provides a fast, low-latency, enterprise-quality (reliable, stable) connection to your Azure resources for cross-premises scenarios.

Unlike a VPN network, traffic does not route (traverse) the internet; it is carried over a private managed network between your resources and Microsoft's data centers; this network is facilitated by a global telecoms carrier, which acts as the connectivity provider.

As expected, this enterprise-grade connectivity has built-in redundancy that uses *dynamic routing* and the **Border Gateway Protocol (BGP)**.

The following diagram outlines the topology of an ExpressRoute circuit connection using a connectivity provider alongside a Site-to-Site VPN connection:

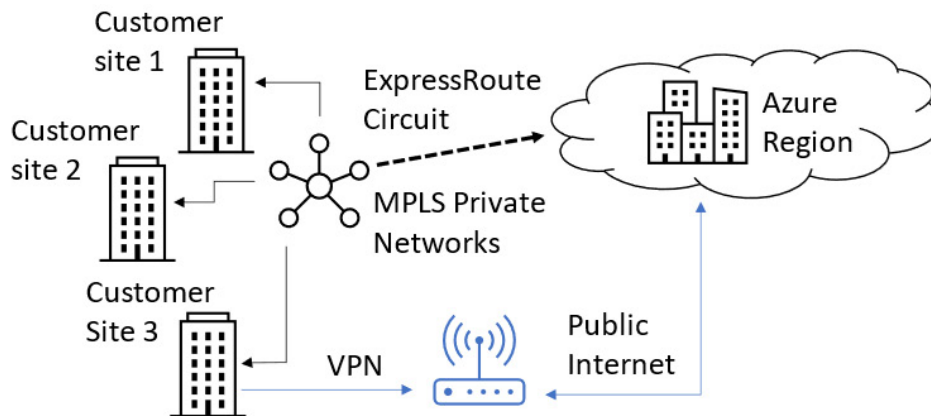


Figure 4.22 – ExpressRoute

As we can see, by working with a connectivity provider, ExpressRoute circuits can make Azure data centers appear just like any other site on an MPLS circuit by extending the on-premises network's reach globally and the data center's capacity.

This section looked at ExpressRoute circuits. In the next section, we will look at Azure storage services.

Azure storage services

Azure provides a range of **storage services** to cater to many differing storage solution needs, each providing different capabilities and features to best match their intended purpose and usage pattern.

You must understand the following *Azure storage services* since they are outlined in the *Describe Core Azure Services* exam:

- Storage accounts, tiers, and replication
- Data stores
- Disk, file, and container (Blob) storage

The content in this section will take your knowledge beyond the exam's objectives so that you are prepared for a real-world, day-to-day Azure-focused role. In the next section, we will look at storage accounts.

Storage accounts

Storage accounts are resources that need to be created to store your data objects; they can be thought of as control panels for your data stores and provide management pane resources.

From a storage account, you can view all your different storage objects, configure security and network settings, perform data management and monitoring, diagnose problems, move and migrate data, apply governance controls such as Azure Advisor recommendations, view activity logs and events, and control access through authentication and authorization.

The different data storage types are categorized into containers, file shares, queues, and tables. Only containers and file shares are part of the exam objectives (at the time of writing); we have only mentioned the other storage types for completeness and so that you can study them, should you wish to.

There are two performance options we can use to create storage accounts, these are Standard and Premium. Premium is recommended for low-latency requirement scenarios, such as sharing user profiles for the Virtual Desktop service, for example.

A storage account also provides a unique namespace (also referred to as an endpoint) for each data object so that each storage account can be addressed uniquely.

The unique identifier for a storage object is constructed in the format of `https://<storage account name>.<service name>.core.windows.net`.

The following are the endpoints of each of the Azure storage services:

- Blob storage: `https://<storage-account>.blob.core.windows.net`
- Data Lake Storage Gen2: `https://<storage-account>.dfs.core.windows.net`
- Azure Files: `https://<storage-account>.file.core.windows.net`
- Queue storage: `https://<storage-account>.queue.core.windows.net`
- Table storage: `https://<storage-account>.table.core.windows.net`

It is also important to understand the following storage service limits (at the time of writing):

- Number of storage accounts per region, per subscription: 250
- Maximum storage account capacity: 5 PiB
- Maximum number of containers, Blobs, file shares, tables, queues, entities, or messages per Storage account: No limit
- Maximum request rate per Storage account: 20,000 requests per second
- Maximum ingress per Storage account (US/Europe regions): 10 Gbps
- Maximum ingress per Storage account (regions other than the US and Europe): 5 Gbps
- Maximum egress for general-purpose v2 and Blob storage accounts (all regions): 50 Gbps

Higher limits for ingress are supported if you send a request to Microsoft regarding increasing your account limits.

This section looked at storage accounts. In the next section, we will look at storage tiers.

Storage tiers

Azure storage services provide different *storage tiers* so that you can store your data objects in the most cost-effective and performant way that meets your needs. The tier you choose should be based on how frequently data is accessed, along with whether data can be moved between tiers at any time; you should also consider the data access/retrieval (rehydration) costs.

The three storage tiers that are available are as follows:

- **Hot tier:** Optimized for frequently accessed data; highest storage costs, lowest access costs.
- **Cool tier:** Optimized for infrequently accessed data that is stored for at least 30 days; lower storage costs, higher access costs.
- **Archive tier:** Optimized for rarely accessed data that is stored for at least 180 days; lowest storage costs, highest access costs, and data retrieval (rehydration) costs; data is offline and it will take several hours for the first byte to be accessible.

This section looked at storage tiers. In the next section, we will look at storage replication.

Storage replication

Microsoft provides redundancy of your data in Azure through several replication options, all of which we will look at in this section. However, note that under the *shared responsibility model*, you are responsible for ensuring your data is available and protected and that you select the most appropriate replication and protection model to meet your needs.

The following are the redundancy options that are available within a primary region, though some regions offer more replication types:

- **Locally redundant storage (LRS)**: Provides three copies of your data, replicated synchronously within a single physical location in the primary region. This is the lowest cost option but also has the lowest availability and durability.
- **Zone-redundant storage (ZRS)**: Provides synchronously replicated copies of your data across three availability zones in the primary region. This is a higher-cost option but it has the highest availability and durability within a region.

The following are the redundancy options that are available within a secondary region:

- **Geo-redundant storage (GRS)**: Provides three copies of your data, replicated synchronously within a single physical location in the primary region; your data is also asynchronously replicated to a single physical location in the secondary region. This secondary region provides three copies of your data, replicated synchronously as LRS.
- **Geo-zone-redundant storage (GZRS)**: Provides synchronously replicated copies of your data across three availability zones in the primary region; your data is also asynchronously replicated to a single physical location in the secondary region. This secondary region provides three copies of your data, replicated synchronously as LRS.

This section looked at storage replication. In the next section, we will look at data stores.

Data stores

Choosing the right **data store** that matches your data type is a key design decision; no single storage solution fits all data types, and several stores might be needed to provide a complete and optimal solution. Your solution will largely be determined by the data type(s) you wish to store.

The key factors in deciding on an optimal storage solution are as follows:

- How you classify your data: structured, semi-structured, unstructured, or streaming; is this relational or non-relational data?
 - a) An example of structured data (also referred to as relational data) would be stored in databases, such as a CRM system. This includes anything with a strict schema, such as most operational data stored in a SQL database or business data stored in a data warehouse for analysis and decision-making.
 - b) Some examples of semi-structured data would be key/value pairs, JSON files, and XML files.
 - c) Some examples of unstructured data would be media files such as photos, videos, audio, and Office documents such as Word documents, PDF files, text files, and log files.
- How your data will be used by the primary operations carried out on each data type: analytical, transactional, read/write, search/lookup, upload, change, and so on.
- How you can get the best performance out of your application.
- How you can get the best durability, availability, recovery, and security.
- How you can be the most cost-effective.

The following diagram provides a simple data store selection guide; for clarity, not all the decision points have been shown:

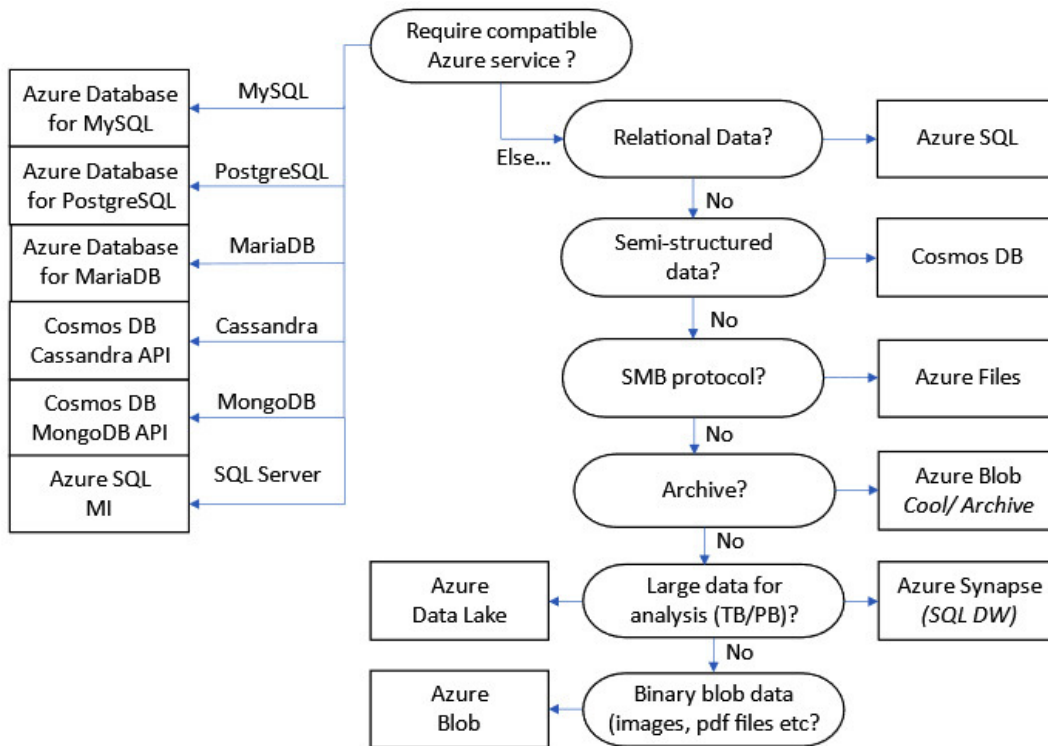


Figure 4.23 – Selecting a data store

This section looked at data stores. In the next section, we will look at Azure Disk Storage.

Disk Storage

Disk Storage provides disks that can be attached to VMs. These provide the OS disk and temp disk that were used in a VM; data disks may also be attached to a VM to provide additional storage capacity and additional volumes to meet disk layout needs, such as a VM that will run SQL Server. In addition, these data disks may be used as an install path for applications or store data that you cannot store on the system volume.

Azure Managed Disks are the recommended disk storage for Azure VMs for persistent data storage. These disks are managed by Microsoft and will not appear in your storage accounts.

The following types of managed disks can be attached to a VM:

- **Standard HDD:** Low-cost storage.
- **Standard SSD:** Consistent performance and low latency.
- **Premium SSD:** High performance and low latency.
- **Ultra disk:** Sub-millisecond latency.

This section looked at disk storage. In the next section, we will look at file storage.

File storage

File storage provides serverless **Server Message Block (SMB)** *file* shares. These are highly available, fully managed network file shares that can be accessed using the traditional SMB protocol and accessed from anywhere, over port 445.

It provides a traditional *mapped drives* storage solution to servers or Windows 10 in a lift-and-shift format.

This section looked at file storage. In the next section, we will look at container (Blob) storage.

Container (Blob) storage

Container storage provides massive-scale, cost-effective storage for unstructured data such as text, video, photos, files, and more.

Containers are used to store **binary large object (Blob)** data objects. For example, the following three forms of Blob data can be stored in a container:

- **Page Blob:** Used to hold random access files such as objects that have been randomly written and read, with no order. An example would be VM disks (managed disks are stored in Microsoft storage accounts).
- **Block Blob:** Used to hold text or binary files, such as objects that have been ordered, are consecutive, and not random. An example would be backups.
- **Append Blob:** Used for logging information, such as objects that have been consecutively added to the last piece of information stored in the Blob. An example would be log files.

This section looked at container storage. In the next section, we will look at the Azure database services.

Azure database services

Database services are one of the core building block services we will be looking at in this chapter.

You must understand the following database services since they are outlined in the *Describe Core Azure Services* exam:

- **Azure SQL Managed Instance (Azure SQL MI):** An Azure-hosted Microsoft SQL Server instance, fully managed as a PaaS service
- **Azure SQL Database:** An Azure-hosted relational database service built on the Microsoft SQL Server database engine; provides single database instances
- **Azure Database for MySQL:** An Azure-hosted relational database service built on the MySQL Community Edition database engine
- **Azure Database for PostgreSQL:** An Azure-hosted relational database service built on the Postgres database engine
- **Cosmos DB:** A NoSQL globally distributed, elastic, and scalable database service

This section introduced the core database services that can be used to build a data-driven solution in Azure. In the next section, we will look at Azure SQL MI.

Azure SQL Managed Instance

Azure SQL MI is a fully managed instance of SQL Server (Enterprise Edition) that's hosted and fully managed by Microsoft as a PaaS service; it is intended to have near full parity with SQL Server instances that you would implement and manage yourself, either on-premises or as IaaS VMs. Think of it as SQL Server as a service.

The following are some of the benefits and capabilities of Azure SQL MI:

- It provides frictionless lift-and-shift scenarios. This is good for new applications or existing legacy applications that want to move to a cloud-hosted solution, require minimal app changes, and wish to take advantage of the latest SQL Server features and database engine.
- It provides a full SQL Server Enterprise Edition database engine instance that's nearly 100% compatible with customer-implemented, owned, and managed SQL Server instances, as would be the case for on-premises deployments or those running on Azure IaaS VMs.
- All the functionality and capabilities of PaaS services are provided, along with IaaS control.
- Full isolation and security.
- VNet implementation and private IP addresses.
- SQL authentication or Azure AD authentication; Windows authentication is not supported.
- Available as a general-purpose and business-critical service tier. It also supports single-instance and instance pools.

This section looked at SQL MI. In the next section, we will look at Azure SQL Database.

Azure SQL Database

Azure SQL Database is a relational database PaaS service that provides Microsoft-hosted, single-instance databases based on the SQL server database engine. Being a PaaS service, Microsoft manages the SQL Server platform, so you don't have to; you can just focus on the database level and the data tasks you wish to perform against it with your data. Think of it as SQL database as a service.

The following are some of the benefits and capabilities of Azure SQL Database:

- It is a fully managed PaaS SQL server platform; you can just focus on the databases and the data.
- It is optimized for modern cloud applications where no legacy requirements or integrations exist.
- It can be used in conjunction with serverless solutions.
- Single databases and elastic pools can be created.
- It is priced on a vCore and a **Data Transaction Unit (DTU)** model, which provides a provisioned and serverless compute tier and the general-purpose, business-critical, and hyperscale service tiers.
- Scale-up and scale-down moving to a different tier is possible; there is no horizontal scaling, which means you cannot provide availability/scale-out to other regions.

This section looked at Azure SQL Database. In the next section, we will look at Azure Database for MySQL.

Azure Database for MySQL

Azure Database for MySQL is a Microsoft-hosted version of the open source MySQL relational database, built on top of the Community Edition and provided as a PaaS service. It is provided as a fully managed database-as-a-service platform; you can focus on just the database level and the data tasks you wish to perform against it with your data.

The following are some of the benefits and capabilities of Azure Database for MySQL:

- Being built on top of Community Edition, there are no compatibility issues with moving databases from on-premises environments.
- It has built-in high availability.
- Data protection is provided, with automatic backups and 35 days of point-in-time restores.
- It is monitored, secure, and up to date and provides automated maintenance for the database's underlying platform hardware, OS, and database engine.
- Data is secured in transit and at rest.
- Three deployment modes are available; Single Server, Flexible Server, and Hyperscale (Citus).
- Elastic scaling is available; that is, horizontal scaling via the Hyperscale (Citus) deployment mode.
- Pay-as-you-go consumption pricing is available through the Basic, General Purpose, and Memory Optimized plans. The prices of these plans rise as additional resources such as CPU and memory are added.

This section looked at Azure Database for MySQL. In the next section, we will look at Azure Database for PostgreSQL.

Azure Database for PostgreSQL

Azure Database for PostgreSQL is a Microsoft hosted version of the open source MySQL relational database but provided as a fully-managed database-as-a-service platform; you can just focus on the database level and the data tasks you wish to perform against it with your data.

The following are some of the benefits and capabilities of Azure Database for PostgreSQL:

- It has built-in high availability.
- Data protection is provided, with automatic backups and 35 days of point-in-time restores.
- It is monitored, secure, and up to date, with automated maintenance of the database's underlying platform hardware, OS, and database engine.
- Data is secured in transit and at rest.
- Three deployment modes are available; that is, Single Server, Flexible Server, and Hyperscale (Citus).
- Elastic scaling is available; that is, horizontal scaling via the Hyperscale (Citus) deployment mode.
- Pay-as-you-go consumption pricing is available through the Basic, General Purpose, and Memory Optimized plans. The prices of these plans rise as additional resources such as CPU and memory are added.

This section looked at Azure Database for PostgreSQL. In the next section, we will look at Cosmos DB.

Cosmos DB

Cosmos DB is a non-relational, NoSQL database service that Microsoft hosts. It is globally distributed and provides horizontally scalable table storage, with built-in redundancy that provides a 99.99% SLA for reads and writes.

This availability capability allows data to be replicated to many Azure regions concurrently. It also provides low latency and performance so that data can be accessed anywhere in the world and scaled seamlessly at the click of a button. Security is also built in and provides row-level authorization and data encryption in transit and at rest.

The following are some of the benefits and capabilities of Cosmos DB:

- It's mission-critical ready, with guaranteed speed at any scale.
- It provides simplified application development.
- It provides use cases for gaming, mobile applications, social applications, retail, marketing, and IoT.
- It can be used in conjunction with serverless solutions such as Azure Functions.
- It is fully managed and cost-effective.
- It can be used to store unstructured and semi-structured data, such as XML and JSON files.
- It is available through the Standard/Autoscale Provisioned Throughput and Serverless Database Operations pricing models, as well as the Single-Region Write (Single-Master) and Multiple-Region Write (Multi-Master) region models.

The following APIs are supported for interacting with Cosmos DB:

- SQL API
- MongoDB API
- Gremlin (Graph) API
- Table API
- Cassandra API

This section looked at Cosmos DB. In the next section, we will look at some hands-on exercises that will reinforce the knowledge you've learned throughout this chapter and increase your skills.

Hands-on exercises

To support your learning with some practical skills, we will learn how to create some of the resources that we looked at in this chapter.

We will look at the following exercises:

- Exercise 1 – creating a VNet
- Exercise 2 – creating a storage account
- Exercise 3 – creating a VM
- Exercise 4 – creating an Azure container instance
- Exercise 5 – creating an Azure web app

Getting started

To get started with these hands-on exercises, you must create a free Azure account at <https://azure.microsoft.com/free>.

This free Azure account provides the following:

- 12 months of free services
- \$200 credit to explore Azure for 30 days
- 25+ services that are always free

Let's move on to the first exercise.

Exercise 1 – creating a VNet

In this section, we will look at how to create a VNet.

Follow these steps to create a VNet:

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type **virtual network** and click on **Virtual Network** from the list of services shown.
3. On the **Virtual networks** blade, click on **+ Create**. It should be at the top of the blade.
4. Set the **Project details** and **Instance details** settings as required via the **Basics** tab.
5. Click **Next: IP Addresses**.
6. On the **IP Addresses** tab, create an IPv4 address space and subnets; you can use the existing defaults or adjust them as required for your scenario. It is recommended to use a NAT gateway for outbound internet access from a subnet; alternatively, you can use an NVA.
7. Click **Next: Security**.
8. On the **Security** tab, choose to enable or disable **BastionHost**, **DDoS Protection Standard**, and **Firewall**, as required for your scenario.
9. Click **Next: Tags** and add any tags as required. Then, click **Next: Review + create**.
10. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
11. You will receive a notification that the resource group was created successfully.
12. From the **Virtual networks** blade, you will be able to see the VNet that was created:

Home >

Virtual networks

milesbetter.solutions (NETORGFT8723201.onmicrosoft.com)

+ Create Manage view Refresh Export to CSV Open query Feedback Assign tags

Filter for any field... Subscription == Project Lobster Resource group == all Location == all Add filter

Showing 1 to 1 of 1 records. No grouping List view

<input type="checkbox"/> Name ↑↓	Resource group ↑↓	Location ↑↓	Subscription ↑↓
<input type="checkbox"/> <=> lob-vnet1-uks	Bisque-RG	UK South	Project Lobster

Figure 4.24 – Virtual networks

In this exercise, we looked at creating a VNet. In the next exercise, we will look at creating a storage account.

Exercise 2 – creating a storage account

In this section, we will look at how to create a storage account.

Follow these steps to create a storage account:

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type **storage account** and click on **Storage accounts** from the list of services shown.
3. On the **Storage accounts** blade, click on **+ Create**. It should be at the top of the blade.
4. On the **Basics** tab, set the **Project details** and **Instance details** settings as required.
5. Click **Next: Advanced**.
6. On the **Advanced** tab, set the **Security**, **Data Lake Storage Gen2**, **Blob**, **Azure Files**, **Tables**, and **Queues** settings as required.
7. Click **Next: Networking**.
8. On the **Networking** tab, set the **Network connectivity** and **Network routing** settings as required.
9. Click **Next: Data Protection**.
10. On the **Data protection** tab, set the **Recovery** and **Tracking** settings as required.
11. Click **Next: Tags** and add any tags as required. Then, click **Next: Review + create**.
12. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
13. You will receive a notification stating that the resource group was created successfully.
14. From the **Storage accounts** blade, you will be able to see the storage account that was created:

[Home](#) >

Storage accounts ✈ ...

✕

milesbetter.solutions (NETORGFT8723201.onmicrosoft.com)

[+](#) Create [⚙](#) Manage view [v](#) [🔄](#) Refresh [↓](#) Export to CSV [🔗](#) Open query [❤](#) Feedback | [🏷](#) Assign tags [🗑](#) Delete

Filter for any field...

Subscription == **Project Lobster**

Resource group == **all** ✕

Location == **all** ✕

[+](#) Add filter

Showing 1 to 1 of 1 records.

No grouping [v](#)

List view [v](#)

<input type="checkbox"/> Name ↑↓	Type ↑↓	Kind ↑↓	Resource group ↑↓	Location ↑↓	Subscription ↑↓	
<input type="checkbox"/> lobsas1uks	Storage account	StorageV2	Bisque-RG	UK South	Project Lobster	...

Figure 4.25 – Storage accounts

In this exercise, we looked at creating a storage account. In the next exercise, we will look at creating a VM.

Exercise 3 – creating a VM

In this section, we will learn how to create a VM.

Follow these steps to create a VM:

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type **virtual machines** and click on **Virtual machines** from the list of services.
3. On the **Virtual machines** blade, click on **+ Create** and then **Virtual machine**. This option should be at the top menu of the blade.
4. On the **Basics** tab, set the **Project details**, **Instance details**, **Administrative accounts**, **Inbound port rules**, and **Licensing** settings as required.
5. Click **Next: Disks**.
6. On the **Disks** tab, set the **Disk options** settings as required; add any **data disks** that are required.
7. Click **Next: Networking**.
8. On the **Networking** tab, set the **Network interface** and **Load balancing** settings as required.
9. Click **Next: Management**.
10. On the **Management** tab, set all the settings that are required.
11. Click **Next: Advanced**.
12. On the **Advanced** tab, set all the settings that are required.
13. Click **Next: Tags** and add any tags that are required. Then, click **Next: Review + create**.
14. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
15. You will receive a notification stating that the resource group was created successfully.
16. From the **Virtual machines** blade, you will be able to see the VM that was created:

Home >

Virtual machines ✈ ...

milesbetter.solutions (NETORGFT8723201.onmicrosoft.com) ✕

[+](#) Create ▾
[↔](#) Switch to classic
 [🕒](#) Reservations ▾
[⚙](#) Manage view ▾
[🔄](#) Refresh
 [↓](#) Export to CSV
 [🔗](#) Open query
 [❤](#) Feedback
 [↔](#) Leave preview ...

Subscription == **Project Lobster**
Resource group == **all** ✕
Location == **all** ✕
+ Add filter

Showing 1 to 1 of 1 records. No grouping ▾ List view ▾


<input type="checkbox"/>	Name ↑↓	Subscription ↑↓	Resource group ↑↓	Location ↑↓	Status ↑↓	Operating system ↑↓	Size ↑↓
<input type="checkbox"/>	 crayfish1-vm	Project Lobster	BISQUE-RG	UK South	Running	Windows	Standard_D2s_v3

Figure 4.26 – Virtual machines

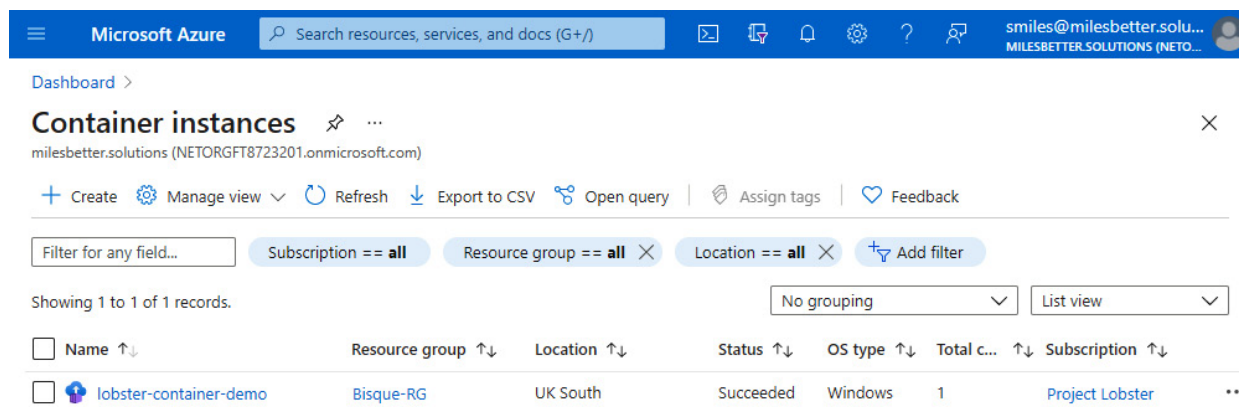
In this exercise, we successfully created a VM. In the next exercise, we will look at creating an Azure container instance.

Exercise 4 – creating an Azure container instance

In this section, we will look at how to create an Azure container instance.

Follow these steps to create a container:

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type **container instance** and click on **Container instances** from the list of services shown.
3. From the **Virtual networks** blade, click on **+ Create**. It should be at the top of the blade.
4. On the **Basics** tab, set the **Project details** and **Container details** settings as required.
5. Click **Next: Networking**.
6. On the **Networking** tab, set the **Networking** settings as required.
7. Click **Next: Advanced**.
8. On the **Advanced** tab, leave the settings as is.
9. Click **Next: Tags** and add any tags as required. Then, click **Next: Review + create**.
10. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
11. You will receive a notification stating that the resource group was created successfully.
12. From the **Container instances** blade, we will be able to see the container that was created:



The screenshot displays the Azure portal interface for the 'Container instances' blade. The top navigation bar includes the Microsoft Azure logo, a search bar, and user information for 'smiles@milesbetter.solu...'. The main content area shows the 'Container instances' page for the resource group 'lobster-container-demo' in the 'Bisque-RG' resource group, located in 'UK South'. The page indicates that 1 record is shown. The table below lists the container instance details.

Name	Resource group	Location	Status	OS type	Total c...	Subscription
lobster-container-demo	Bisque-RG	UK South	Succeeded	Windows	1	Project Lobster

Figure 4.27 – Container instance

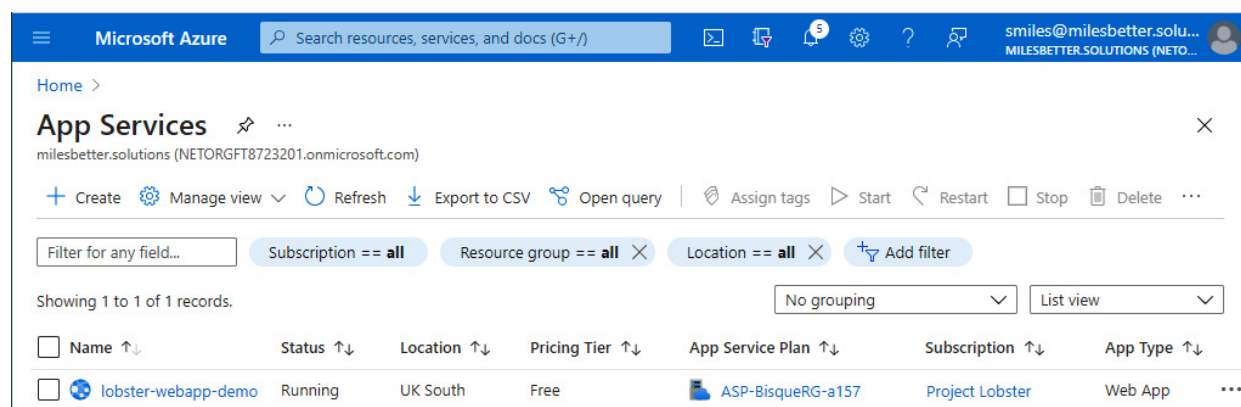
In this exercise, we looked at creating an Azure container instance. In the next exercise, we will look at creating a web app.

Exercise 5 – creating an Azure web app

In this section, we will look at the steps to create a web app.

Follow these steps to create a web app:

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.
2. In the search bar, type **app service** (or **web app**) and click on **App Services** from the list of services.
3. From the **Virtual machines** blade, click on **+ Create** and then **Virtual machine**. This option should be at the top menu of the blade.
4. From the **Basics** tab, set the **Project Details** and **Instance Details** settings as required. Then, set **App Service Plan** to use an existing one or create new App Service plan, and set the size as required.
5. Click **Next: Deployment**.
6. From the **Deployment** tab, leave the settings as is.
7. Click **Next: Monitoring**.
8. From the **Monitoring** tab, enable **Application Insights**.
9. Click **Next: Tags** and add any tags as required. Then, click **Next: Review + create**.
10. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
11. You will receive a notification stating that the resource group was created successfully.
12. From the **App Services** blade, you can see the web app that was created:



The screenshot shows the Microsoft Azure portal interface. At the top, there is a navigation bar with the Microsoft Azure logo, a search bar, and user information for 'smiles@milesbetter.solu...'. Below the navigation bar, the 'App Services' blade is open, displaying a list of web apps. The list has columns for Name, Status, Location, Pricing Tier, App Service Plan, Subscription, and App Type. One web app is listed: 'lobster-webapp-demo' with a status of 'Running', located in 'UK South', on a 'Free' pricing tier, using the 'ASP-BisqueRG-a157' App Service Plan, under the 'Project Lobster' subscription, and is a 'Web App' type.

Name	Status	Location	Pricing Tier	App Service Plan	Subscription	App Type
lobster-webapp-demo	Running	UK South	Free	ASP-BisqueRG-a157	Project Lobster	Web App

Figure 4.28 – App Services

In this exercise, we successfully created an App Service web app.

This section covered hands-on labs. Now, let's summarize this chapter.

Summary

This chapter provided complete coverage of the *Describe Core Azure Services* AZ-900 Azure Fundamentals exam skills area.

In this chapter, you learned about various skills that will provide you with the confidence to explain and discuss numerous aspects of the core Azure services with a business or technical audience. This includes describing the benefits and usage of Azure Marketplace, as well as Azure resources such as compute, storage, network, and data.

Knowledge beyond the exam's content was also provided to help you prepare for a real-world, day-to-day Azure-focused role.

The chapter concluded by providing some hands-on exercises that brought all the skill areas that were covered in this chapter together. The next chapter will outline the core resources available in Azure, such as compute, network, storage, and data. We will also look at the Azure Marketplace.

Additional information and study references

The following are some links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure Fundamentals:
<https://docs.microsoft.com/learn/certifications/exams/az-900>
- Exam AZ-900: skills outlined:
<https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- Microsoft Learn: Azure Fundamentals – Describe Core Azure Services:
<https://docs.microsoft.com/learn/modules/azure-architecture-fundamentals>

Skill check

Challenge yourself regarding what you have learned in this chapter:

1. Explain what an Azure resource is.
2. Explain the Azure Marketplace and how it can be used.
3. Define the term *compute*.
4. List four Azure compute services.
5. When are VMs the best choice for a compute service?
6. When using VMs, what are you still responsible for?
7. Explain the differences between a traditional physical server approach and virtualization for delivering applications.
8. Explain the different VM *family* series.
9. List four additional VM resources that should be considered.
10. List a minimum of five additional elements to consider when deploying VMs into a solution.
11. Explain the difference between the *virtualization* and *containerization* approaches.
12. Explain the difference between *ACI* and *AKS*.
13. Explain *Azure App Service* and how it differs from VMs.
14. Explain the *Virtual Desktop* service.
15. Explain what Azure Virtual Network is.
16. Explain VPNs and ExpressRoute.
17. Explain a storage account and storage tiers.
18. Explain storage replication.
19. What is the most appropriate data store for an SMB file share?
20. List five different database services and their different use cases.

Section 3: Core Solutions and Management Tools

This section provides complete coverage of the knowledge and skills required for the *skills measured* of the *Core solutions and management tools* section of the exam. We also cover knowledge and skills that go beyond the exam content, so you are prepared for a real-world, day-to-day Azure-focused role.

This part of the book comprises the following chapters:

- [Chapter 5](#), *Core Azure Solutions*
- [Chapter 6](#), *Azure Management Tools*

Chapter 5: Core Azure Solutions

In [Chapter 4](#), *Core Azure Resources*, you learned about **compute**, **storage**, **networking**, **databases**, and the **Azure Marketplace**.

This chapter will outline the *core solutions* available in Azure, including **serverless computing**, **artificial intelligence**, **Internet of Things**, **big data** and **analytics**, and **DevOps**.

This chapter aims to provide complete coverage of the AZ-900 Azure Fundamentals *Skills Measured* section known as *Describe Core solutions on Azure*.

By the end of this chapter, you will have learned about the following skills:

- Serverless computing solutions, including Azure Functions and Logic Apps
- Azure Machine Learning, Cognitive Services, and Azure Bot Service
- Internet of Things (IoT) Hub, IoT Central, and Azure Sphere
- Azure Synapse Analytics, HDInsight, and Azure Databricks
- Azure DevOps, GitHub, GitHub Actions, and Azure DevTest Labs

To support your learning with some practical skills, we will also look at how to create some of the resources covered in this chapter.

To do so, we will perform the following exercises:

- Exercise 1 – Creating a serverless solution using an Azure Function
- Exercise 2 – Creating a serverless solution using an Azure Logic App
- Exercise 3 – Creating an IoT solution using an Azure IoT Hub
- Exercise 4 – Creating an AI solution using a Bot Service

In addition, this chapter's goal is to take your knowledge beyond the exam's content so that you are prepared for a real-world, day-to-day Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will need the following:

- An Azure subscription that can create and delete resources in the subscription. If you do not have an Azure subscription, you can create a free Azure account by going to <https://azure.microsoft.com/free>.
- Access to an internet browser so that you can log into the Azure portal: <https://portal.azure.com>.
- Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.
- A Twitter account and an Outlook account for sending notifications (<https://help.twitter.com/en/using-twitter/create-twitter-account> and <https://www.microsoft.com/en-gb/microsoft-365/outlook/>, respectively).

Serverless computing solutions

This section will cover the following exam objective: *describe the benefits and usage of serverless computing solutions, including Azure Functions and Logic Apps.*

Before we look closer at the serverless computing solutions that are part of the exam objectives, we will create a knowledge foundation and baseline to build from. This also aims to build an understanding of the bigger picture of how the different *compute solutions* are positioned and interrelated for both *technical* and *business* personas.

As we learned in [Chapter 1, Introduction to Cloud Computing](#), serverless computing arose due to the evolution of cloud computing platforms and is an architectural shift in the *compute layer*. This evolution moves IT professionals in a new direction, away from *infrastructure* in the deployment, scale, and management layers and toward the *business logic* layer. This is the new unit of deployment, scale, and management; it extends and evolves the *PaaS* cloud computing services model.

The term *serverless* is a *misnomer*; there are still servers and an infrastructure to execute your code or workflow on. The concept is about *not needing* to create, manage, or pay for the underlying infrastructure so that you can execute your code or workflow. The term *serverless* just means that Microsoft *abstracts* this entirely (*removes it from your concern and control*) and provides this cloud-hosted serverless code and workflow execution environment to the customer *as a service*.

In *serverless*, we say that the *runtime and languages are abstracted*; when we say abstracted, we mean that this requirement is removed from us and provided back to us as a service by the platform provider. When we compare serverless to the other Azure platform compute services, we can say that *PaaS* is about *abstracting the compute*, while *IaaS* is about *abstracting the hardware*. The following diagram visualizes the serverless compute service's position:

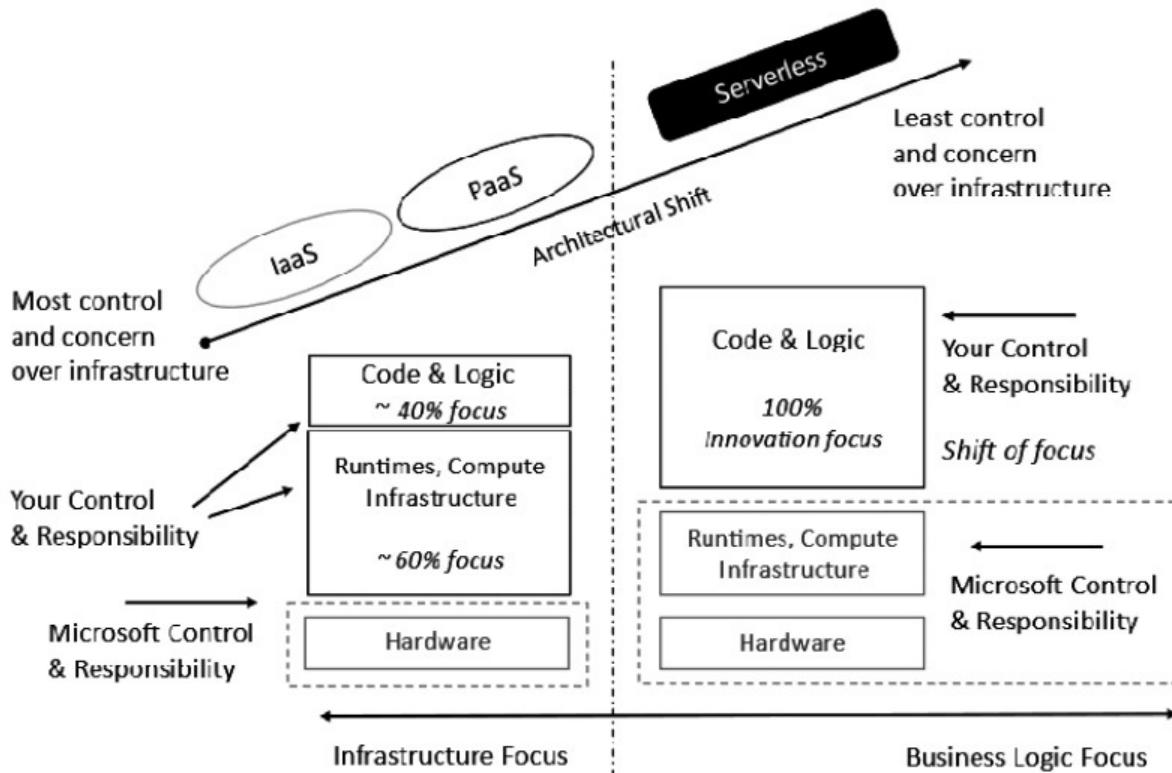


Figure 5.1 – Serverless positioning

In the preceding diagram, we can see that our focus has shifted away from infrastructure, compute, and runtimes so that you can focus on the *code and logic*; the lower layer's *needs* for runtimes, compute, and infrastructure are not our *concern* and not under our *control*; they have been *abstracted*. You can now focus your effort on the code and logic. The serverless approach, referred to as *cloud-native*, is a modern and new way of thinking and adopting the cloud mindset we looked at in [Chapter 1, Introduction to Cloud Computing](#). This requires applications and solutions to be rebuilt and rearchitected to take advantage of these cloud-native benefits. This means that it won't be a suitable answer to some questions and is best suited to event-based solutions. However, it provides another way that applications and solutions can be delivered to best suit the demands of an organization.

The following are the key takeaways to answer why serverless may be chosen as the compute service for a workload:

- Reduced Development and Operations resources and their costs (*time to deliver software and systems using a DevOps culture*)
- Reduced time to market

- Per-action billing

To cover the *exam objectives*, we will look at the following two components of the Azure serverless category:

- **Azure Functions:** Serverless *compute* service
- **Azure Logic Apps:** Serverless *workflow* service

This section introduced serverless solutions. In the next section, we will look at Azure Functions.

Azure Functions

Azure Functions is an *event-triggered serverless compute service* that's provided as a cloud-hosted development *Platform-as-a-Service (PaaS)*.

In a nutshell, this means that an event such as uploading a file, a web request (*HTTP trigger*), a schedule being met, and so on can be used as a *trigger* to execute small pieces of code; we call this a *code-first (imperative)* development approach.

Furthermore, adopting Azure Functions means that there is only the need to focus on the *business logic* and the task that needs to be achieved; there is no need for any applications, runtimes, or infrastructure to be specified. The following diagram visualizes the Azure Functions components:

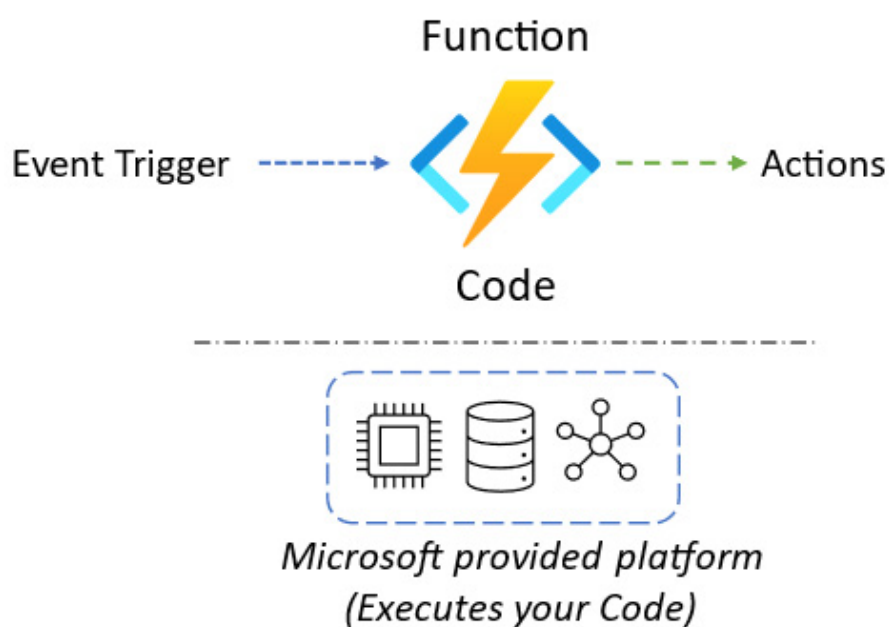


Figure 5.2 – Serverless code execution platform

The key takeaway from the preceding diagram is that an *event* triggers *code*; we will look at this in the next section for Azure Logic Apps.

The benefits of Azure functions are that they make development efforts more productive, agile, and efficient; we can focus on creating multiple single-purpose action functions (*pieces of code*) that are lightweight and less complex than a traditional

monolithic approach to executing code. Functions support most common development languages, such as C#, F#, Node.js, Python, PHP, and so on.

The following methods can be used to create functions:

- The Azure portal
- Microsoft Visual Studio and Visual Studio Code
- The Azure *Command-Line Interface (CLI)*

We will look at *Logic Apps* in the next section, but we can think of Azure Logic Apps as an orchestrator and workflow engine; it can orchestrate and call many different functions, each performing a task as part of the bigger solution.

Azure Logic Apps

Azure Logic Apps is an *event-triggered serverless workflow (and orchestration) service* that's provided as a cloud-hosted development PaaS. It models *business processes* in *automated workflows*.

In a nutshell, this means that it reacts to an event much like Azure Functions does, but instead of *triggering code execution*, Azure Logic Apps *triggers a workflow*. A workflow is a set of conditional task actions (*operations*) and can act as an integration orchestration service for these business processes.

This is classed as a *designer-first (declarative)* development approach; there is *low code* or *no code*, as is the case with Functions. We create the orchestration workflows by drawing out the business process as a flow diagram; we create these business processes visually using a Logic Apps designer in the Azure portal to create the workflow. We will cover this in more detail in the hands-on exercises.

Code can be still used in Logic Apps as part of the flow by using inline code or by adding Azure functions inside the Logic App's flow.

These business process workflows are created from an extensive collection of ready-made connectors that perform the workflows tasks. An operation that performs a task is called an *action* and follows an *if, then* approach, which means that when the Logic App receives a *trigger*, an *action* (operation) can be performed.

It is possible to chain a set of functions as *actions* through a workflow based on an event and a trigger. An event trigger could be when a Twitter message is received when a user is added to an internal HR system or when a file is uploaded/downloaded for SharePoint, blob or file storage, other business applications, and so on.

The difference from a *function* is that Logic Apps is a *no-code (designer-first)* model; the triggers execute a workflow operation action (*business process instead of code*) that could send an email, post a team's message, post a tweet, copy data to a data store, update a CRM system, call a function, and so on; there can be multiple operation actions for the workflow. This means Logic Apps acts as an automated workflow engine, an orchestrator of functions, tasks, and business processes; it allows integrations between systems, services, and data. Logic Apps workflows are created in

the Azure portal through a visual designer interface; no coding is required. The following diagram visualizes the Azure Logic Apps components:

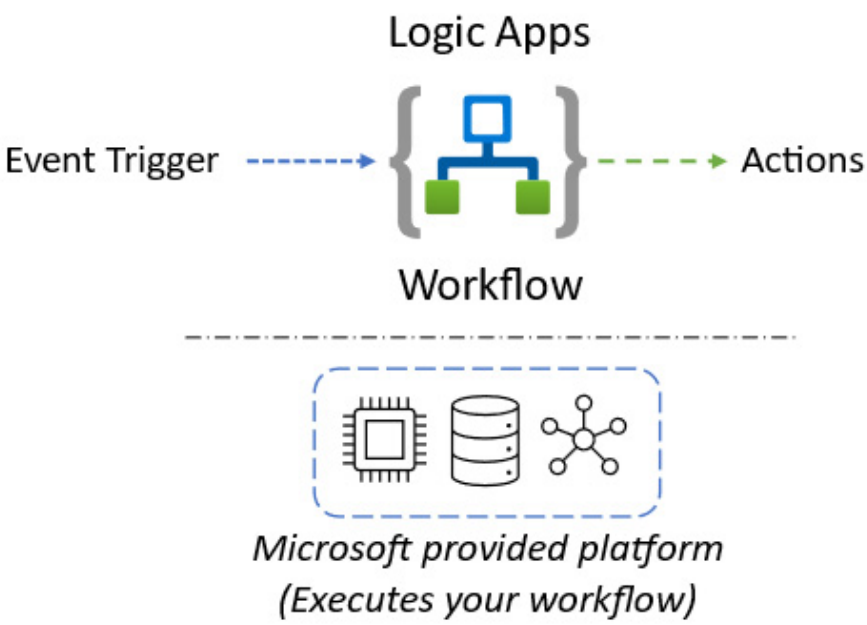


Figure 5.3 – Serverless workflow/orchestration platform

The key takeaway from the preceding diagram is that an *event* triggers a *workflow*, a set of operations known as actions; this can be contrasted with Azure Functions, where *events* trigger *code*.

The following diagram visualizes a use case scenario for serverless using Functions and Logic Apps in the solution:

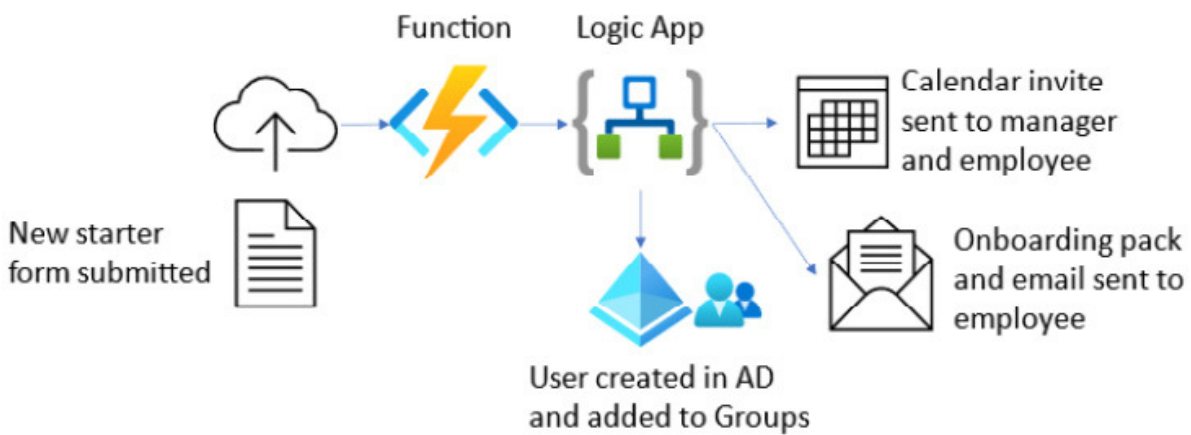


Figure 5.4 – Serverless use case scenario

This section looked at Azure Logic Apps. In the next section, we will look at AI solutions.

Artificial intelligence solutions

This section will aim to cover the following exam objective: *describe the benefits and usage of Azure Machine Learning, Cognitive Services, and Azure Bot Service.*

Before we look closer at the *Microsoft AI platforms* that are part of the exam objectives, we will create a knowledge foundation and baseline to build from. This also aims to build an understanding of the bigger picture of how the different *AI techniques* are positioned and interrelated for both *technical* and *business* personas.

Artificial Intelligence (AI) is the ability of a computer to imitate intelligent human behavior. AI allows computers to make predictions based on data, analyze images, take actions, recognize speech and text, and interact naturally.

The following are some important terminologies to understand regarding AI:

- **Artificial Intelligence (AI):** This is the broad term given to describe the ability to mimic human intelligence by a computer; it can be applied to any app that reasons, senses, acts, and adapts behaviors or outcomes.
- **Machine Learning (ML):** This is the ability for computers to improve at tasks through experience (*learning*); this is known as *training* and it will produce a *model* that can be *deployed* and is said to be *trained*. ML is based on *algorithms*, whose output improves over time as they are fed more data to analyze and learn from; these are known as *training datasets*. It is a subset of *AI*.
- **Deep Learning (DL):** This is the ability for a computer to train itself to perform a task; this is based on multi-layered neural networks and they can achieve this self-learning from vast amounts of data. It is a subset of *ML*.

The following diagram illustrates AI, ML, and DL:

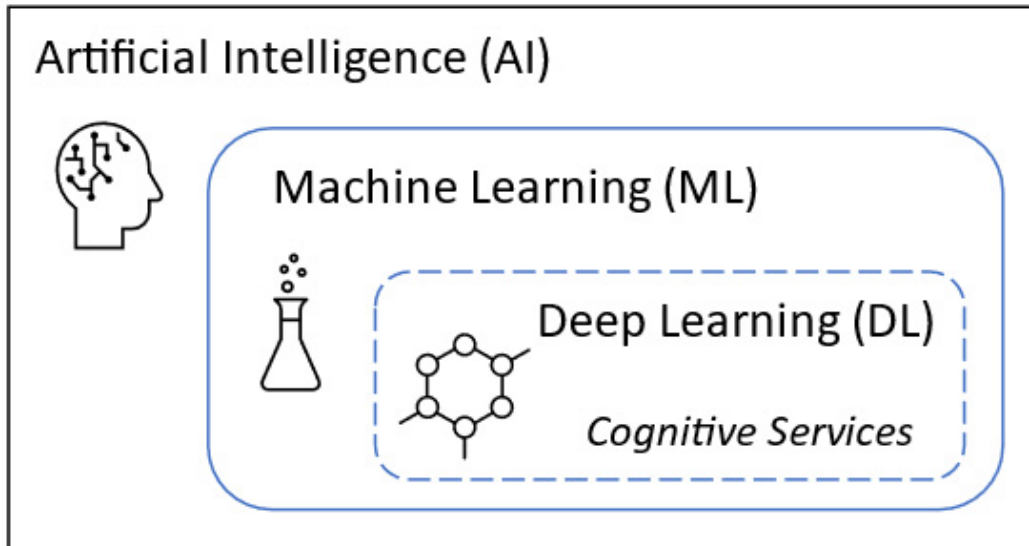


Figure 5.5 – Artificial intelligence inter-relations

No matter which approach we take to creating AI, there are ethical and legal considerations we must understand; there needs to be a governance framework that controls how AI is created and used. Microsoft has six *AI principles*: **Fairness, Reliability and Safety, Privacy and Security, Inclusiveness, Transparency, and Accountability.**

The core concept of AI is based on *algorithms*; this can be thought of as the *magic sauce*, as it were: an algorithm is a way to *solve a problem* or carry out *dataset analysis* using a sequence of *calculations* and *rules*.

For the *exam objectives*, we will look at the following three Microsoft AI platform services:

- **Azure Machine Learning:** A custom AI service
- **Azure Cognitive Services:** A pre-built AI service
- **Azure Bot Service:** A conversational AI service

The following diagram positions these platform services into their *machine learning* and *deep learning* subsets of *artificial intelligence*:

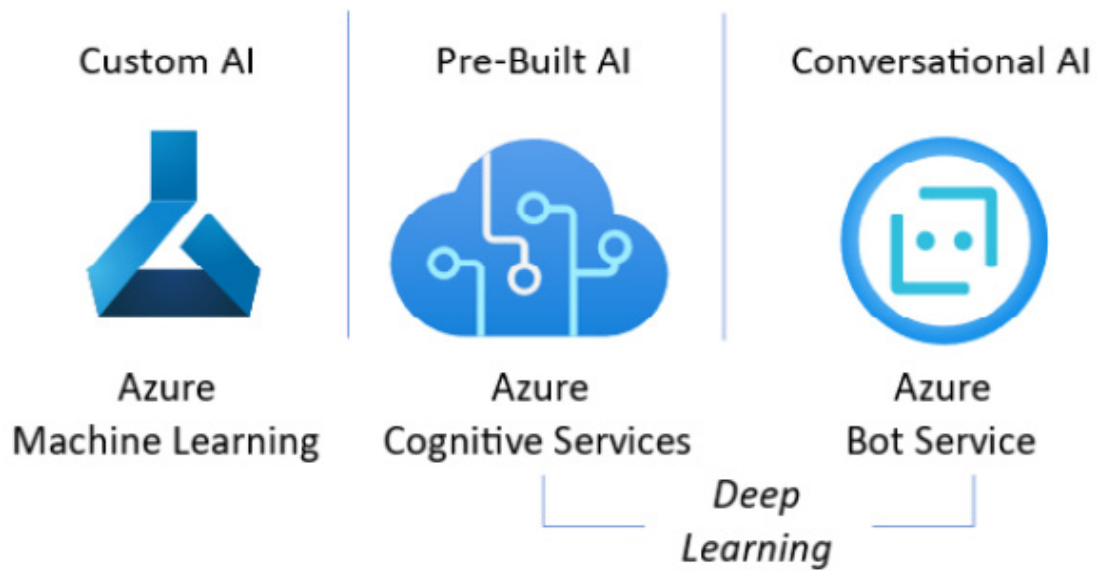


Figure 5.6 – Microsoft AI platform services

This section introduced AI. In the next few sections, we will look at the various Microsoft AI platform services we can use. First, we will look at the Azure Machine Learning service.

Azure Machine Learning

In a nutshell, **Azure Machine Learning (ML)** is Microsoft's fully managed cloud-based ML environment for making predictions from data. It is comprised of a collection of services and tools that allow you to train, deploy, and manage ML models at an enterprise scale.

The value of ML to an organization is the ability to predict future trends or behavior; Azure ML's capabilities allow these predictions to be made better, faster, simpler, and with a lower cost barrier.

The Azure ML service provides capabilities for building and deploying models while having complete control of the design of the algorithm, training, and your data.

The models can be trained and deployed at scale using web interfaces and *Software Development Kits (SDKs)*. In addition, open source technologies can be used, including Python frameworks such as PyTorch, TensorFlow, and scikit-learn; Azure ML also supports Python and R. The following diagram outlines an Azure ML solution approach:

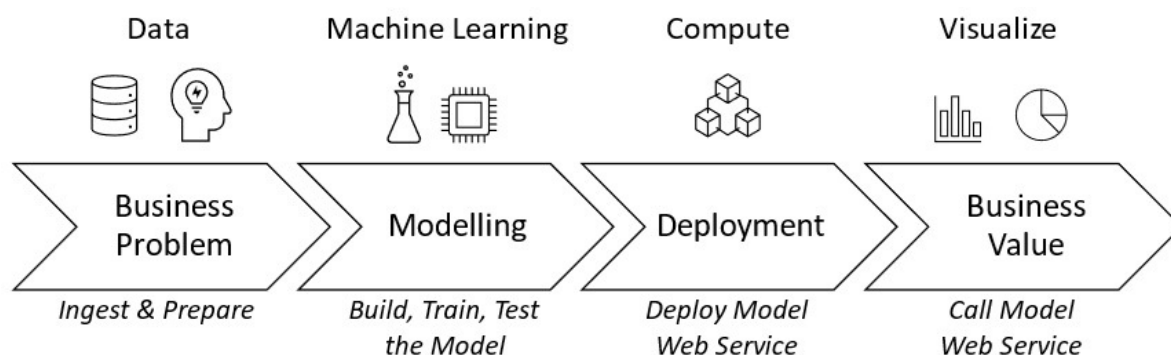


Figure 5.7 – Azure Machine Learning solution approach

As we learned in the *Artificial intelligence solutions* section, ML is a *data science* technique and a subset of AI.

The most common use cases for ML within AI are image analysis, natural interaction, speech comprehension, and making predictions from data.

The core concept of ML is based on *algorithms*; this can be thought of as the *magic sauce*, as it were. An algorithm is a way to solve a problem or carry out dataset

analysis using a sequence of calculations and rules. For example, an algorithm, in its simplest form, can determine what similar category objects should be classified. The following figure shows an example of where an algorithm would be used to figure out what's a dog or a muffin when classifying these images:

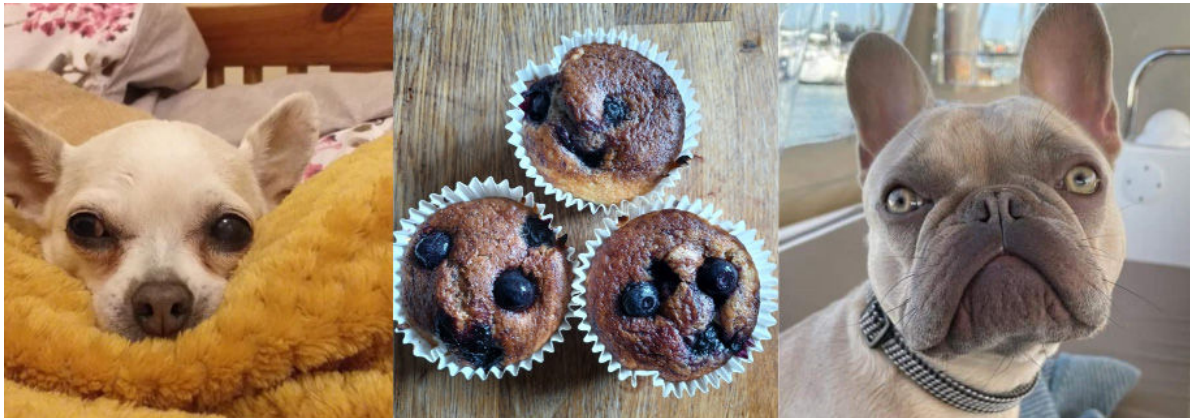


Figure 5.8 – Sadie, Mrs Smile's Muffins, or Daisy

This section looked at Azure Machine Learning. In the next section, we will look at Azure Cognitive Services.

Azure Cognitive Services

Azure Cognitive Services is a cloud-based pre-built AI and is a fully managed Microsoft platform; it simplifies and speeds up the creation and deployment of AI solutions.

Azure Cognitive Services provides pre-built ML models that allow developers to enable *speech, hearing, understanding, and reasoning* in their apps while also bringing in live data. This differs from Azure ML, which is based on requiring you to create models trained on your data.

These pre-built models can be accessed by developers via the Cognitive Services API; this functionality can be added with some lines of code. This is a developer skillset and in contrast to the Azure ML service, you do not require data science or specific ML or language/framework knowledge to get value from these services; think of this as an ML-as-a-Service approach.

Azure Cognitive Services is comprised of the following:

- **Vision:** This service provides identification and recognition capabilities for analyzing pictures, video, and other visual content.
- **Speech:** This service provides text-to-speech and speech-to-text conversion, language translation, and speaker recognition and verification.
- **Language:** This service provides pre-built scripts for processing natural language, key phrase extraction, sentiment analysis, and user recognition.
- **Decision:** This service provides personal recommendations, content moderation, and abnormality detection.

This section looked at Azure Cognitive Services. In the next section, we will look at Azure Bot Service.

Azure Bot Service

Azure Bot Service is a cloud-based, fully managed platform that allows you to create intelligent enterprise-grade bots. Essentially, a *bot* is a software program that is designed to perform a particular automated task. Bots use AI to learn human activity and behaviors and interact through speech and text. The most common example of this is a chatbot, which uses conversational AI to help engage with customer service scenarios.

Azure Bot Service is provided as a PaaS service so that it can be quickly adopted with no need to create any underlying compute or infrastructure. Being built on top of Azure App Service, it has built-in capabilities regarding portal UI-driven simplicity, scale, and automation.

Web App Bot templates can be used to get started; you select the one that best meets your needs. The SDK language can be either C# or Node.js.

All this happens/is orchestrated for you in what is known as the *Bot Framework*; this streamlines the portal experience and allows you to build, customize, edit, and download your source code, view analytics, and so on. You can even do in-portal testing by asking your bot a question directly within the creation environment to see what response you get.

This section looked at Azure Bot Service. In the next section, we will look at Internet of Things solutions.

Internet of Things solutions

This section will aim to cover the following exam objective: *describe the benefits and usage of Internet of Things (IoT) Hub, IoT Central, and Azure Sphere.*

Before we look closer at the *IoT solutions* that are part of the exam objectives, we will create a knowledge foundation and baseline to build from. This also aims to build an understanding of the bigger picture of how the different *IoT solutions* are positioned and interrelated for both *technical* and *business* personas.

IoT is a technology solution that provides intelligent devices (*things*) equipped with sensors to collect and send data to a cloud platform for analysis and take action based on insights.

The three core elements of an IoT solution are *collecting the data*, *processing the data*, and *taking action on the data*. You need a set of connected technologies across these three areas, which are outlined as follows:

- **Things:** The *physical things* that have embedded sensors that, when connected to the internet, can send telemetry (*values*) data to a cloud platform for *analysis* and *action*.
- **Insights:** The results from analyzing the received data (*values*) from the *things*. These insights are produced by real-time stream analysis, machine learning, and other backend processes.
- **Actions:** The responses to the insights. These can be manual or automated. For example, the actions could be to change the device's settings, update an inventory or metrics dashboard, trigger an intervention such as scheduling a site visit/appointment with a professional, sending a part, ordering an inventory item, and so on.

The following diagram visualizes these core IoT elements:

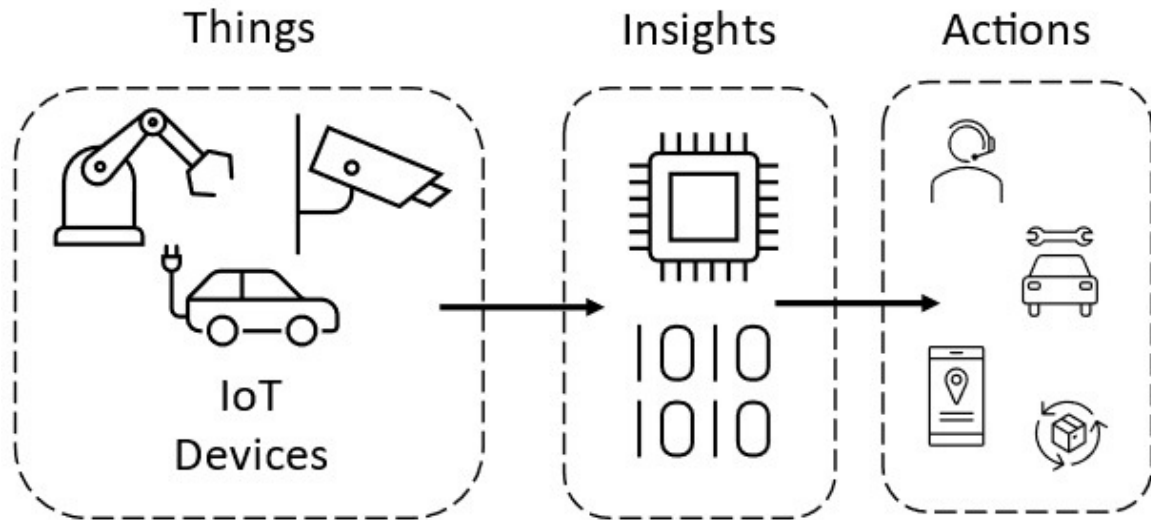


Figure 5.9 – Core IoT elements

IoT solutions have many use case scenarios across many industries. The following are the most common scenario categories:

- **Remote monitoring:** The foundation service for all IoT solutions; telemetry data is sent to the cloud for monitoring devices with embedded sensors.
- **Predictive maintenance:** This determines when maintenance should be done based on receiving values on the condition of in-service equipment from device sensors.
- **Facilities management:** This allows you to optimize energy consumption, control access, and the user experience.
- **Connected manufacturing:** Here, embedded IoT devices and sensors are used to manufacture production lines or similar.
- **Fleet management:** Here, location-aware sensors are fitted to monitor a fleet of vehicles, such as trucks, cars, forklifts, drones, ships, containers, planes, trains, and so on.

This section introduced the high-level concepts of IoT. In the next section, we will look at IoT solutions.

Azure IoT solutions

Azure IoT has a collection of solutions that allow an organization to take actions based on valuable insights that have been captured from data that's been received from connected devices. These solutions can help address the most common IoT challenges of *cost*, *complexity*, and *security*.

The Azure IoT technology portfolio allows computational sensor devices (*things*) to send recorded values (*transmit telemetry*) to the Azure computing platform for processing. This is where actions can be taken on data that's been received and visualizations that have been presented; a device may also be set to receive desired values to adjust its settings so that they're in line with the received values. The following diagram shows an Azure IoT reference architecture:

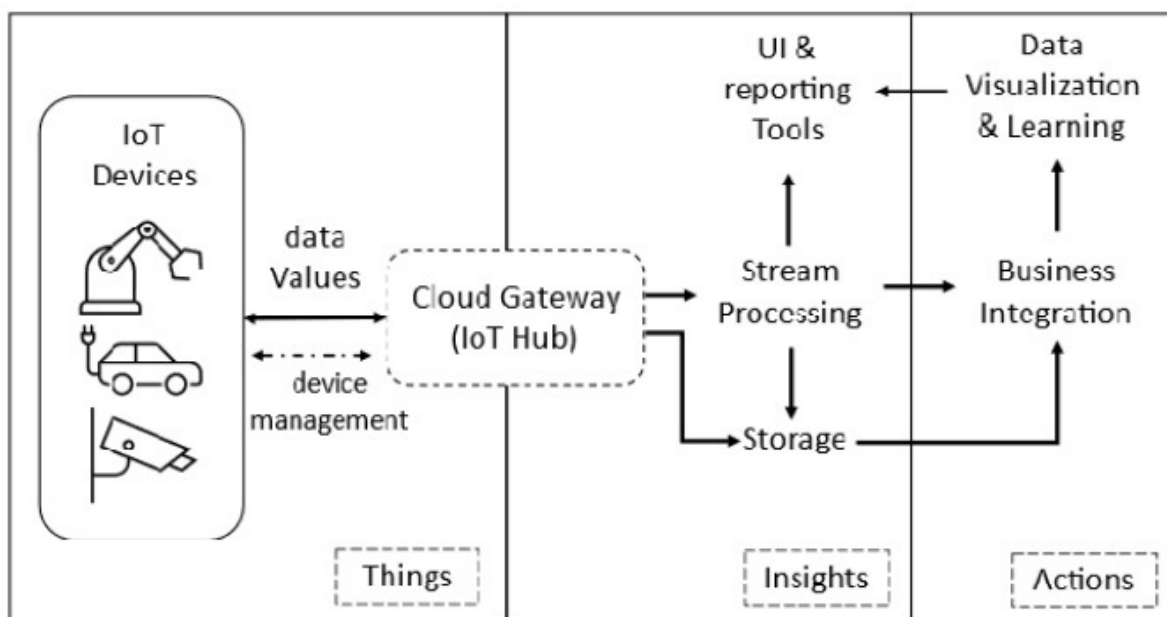


Figure 5.10 – Azure IoT reference architecture

The preceding reference architecture is made up of the following components:

- **Devices:** These register with the Azure IoT platform and send telemetry data. These devices, in some scenarios, might send data locally to an edge gateway device; an edge device performs some local processing optimizations.
- **Cloud gateway:** The cloud gateway reads the ingested data (received) securely and provides device management capabilities.

- **Stream processors:** These consume the telemetry data that's sent by the gateway. They integrate with the business processes and will take action based on the insights; they place data into storage.
- **Visualization interface:** Users will interact with this interface to visualize the data; it provides easy device management.

We will look at the following three core components of the Azure IoT services portfolio for the exam objectives:

- **Azure IoT Central:** Consume IoT services.
- **Azure IoT Hub:** Build your own IoT services.
- **Azure Sphere:** Secure IoT services.

This section introduced various Azure IoT solutions. In the next section, we will look at Azure IoT Central.

Azure IoT Central

Azure IoT Central is a SaaS fully managed IoT app platform. It allows you to rapidly adopt IoT solutions through its ease to connect to data for valuable insights and operate and manage IoT devices at scale.

Choosing to use IoT Central as your core IoT service will be appropriate when you need to quickly build IoT applications through a fully managed IoT solution offering. Here, you can align with the following:

- **Management:** You need a fully managed IoT application platform that will handle scale, security, and management so that you don't have to. You can think of this as IoT-as-a-Service.
- **Control:** You wish to control user admin customizations and your telemetry data (management and data plane), but you do not wish to have the overhead of managing the underlying IoT system and how it's managed.
- **Pricing:** You need a simple and predictable pricing structure.

This section looked at IoT Central. In the next section, we will look at Azure IoT Hub.

Azure IoT Hub

Azure IoT Hub is a PaaS cloud-hosted Azure solution that provides a managed connectivity service. You can use this to build a network to connect and collect insights from virtually any number of sensor-fitted devices at a massive scale (*one million devices per hub*). It also allows developers to build customized device monitoring and management solutions. It acts as a bi-directional communication channel and message

hub between the devices and the IoT applications. Azure IoT Hub can integrate with Azure Logic Apps, Azure Machine Learning, and Azure Stream Analytics.

Choosing to use IoT Hub as your core service will be appropriate when you need an IoT solution offering that can provide a building block approach to building customized and complex IoT scenarios. It aligns with the following:

- **Management:** You need full control of the underlying IoT systems and services that make up the solution. This includes being in control of scaling to meet needs and connecting devices to the solution, managing updates and settings on the devices, and providing end-to-end security.
- **Control:** You require total customization for all the aspects of the solution architecture.
- **Pricing:** You wish to be able to control costs by optimizing your services.

This section looked at IoT Hub. In the next section, we will look, at Azure Sphere.

Azure Sphere

Azure Sphere is an end-to-end IoT security solution where the security of telemetry data and communications is critical.

There are three components to Azure Sphere:

- **Hardware:** This is the Azure Sphere *microcontroller unit (MCU)*, which hosts the OS and receives the sensor data.
- **OS:** This is a customized Linux OS and runs the security service.
- **Security Service:** This ensures that the device has not been compromised. The device must always authenticate before sending data. The security service checks that the device has not been tampered with before providing a secure communication channel that can push any updates to the device. Once Azure Sphere has validated the device and ensured any updates are available, it allows the device to send data and receive instructions for the settings that need to be updated.

An example would be in the case of financial scenarios such as ATM devices; it encompasses the device's hardware, OS, and the secure communication channel to send data and push updates where security patching is required.

This section looked at Azure Sphere. In the next section, we will look at big data and analytics solutions.

Big data and analytics solutions

This section will aim to cover the following exam objective: *describe the benefits and usage of Azure Synapse Analytics, HDInsight, and Azure Databricks.*

Before we look closer at the **big data and analytics solutions** that are part of the exam objectives, we will create a knowledge foundation and baseline to build from. This also aims to build an understanding of the bigger picture of how the different *big data and analytics solutions* are positioned and interrelated for *technical* and *business* personas.

We should first understand what we mean by *big data and analytics*. In a nutshell, it is about discovering information hidden in data, which should present actionable/acknowledgeable information to help an organization make informed decisions and, dependent on the context, gain a competitive advantage.

The challenge is that traditional *Data Warehouse (DW)* solutions and technologies cannot handle the massive volumes of complex, unstructured data, which is a characteristic of big data. This makes the traditional DW approach defunct due to its cloud mindset.

Now that we understand what big data and analytics are, what are some of their use cases and scenarios?

There are three typical scenarios where we can utilize big data to arrive at better outcomes:

- Modern Data Warehouses
- Intelligent analytics
- IoT

There are four types of analytics techniques for getting insights out of data; some are based on traditional *business intelligence (BI)* analysis techniques, while others are based on *AI (machine learning)* techniques. The following diagram visualizes these analytics techniques and their relationships:

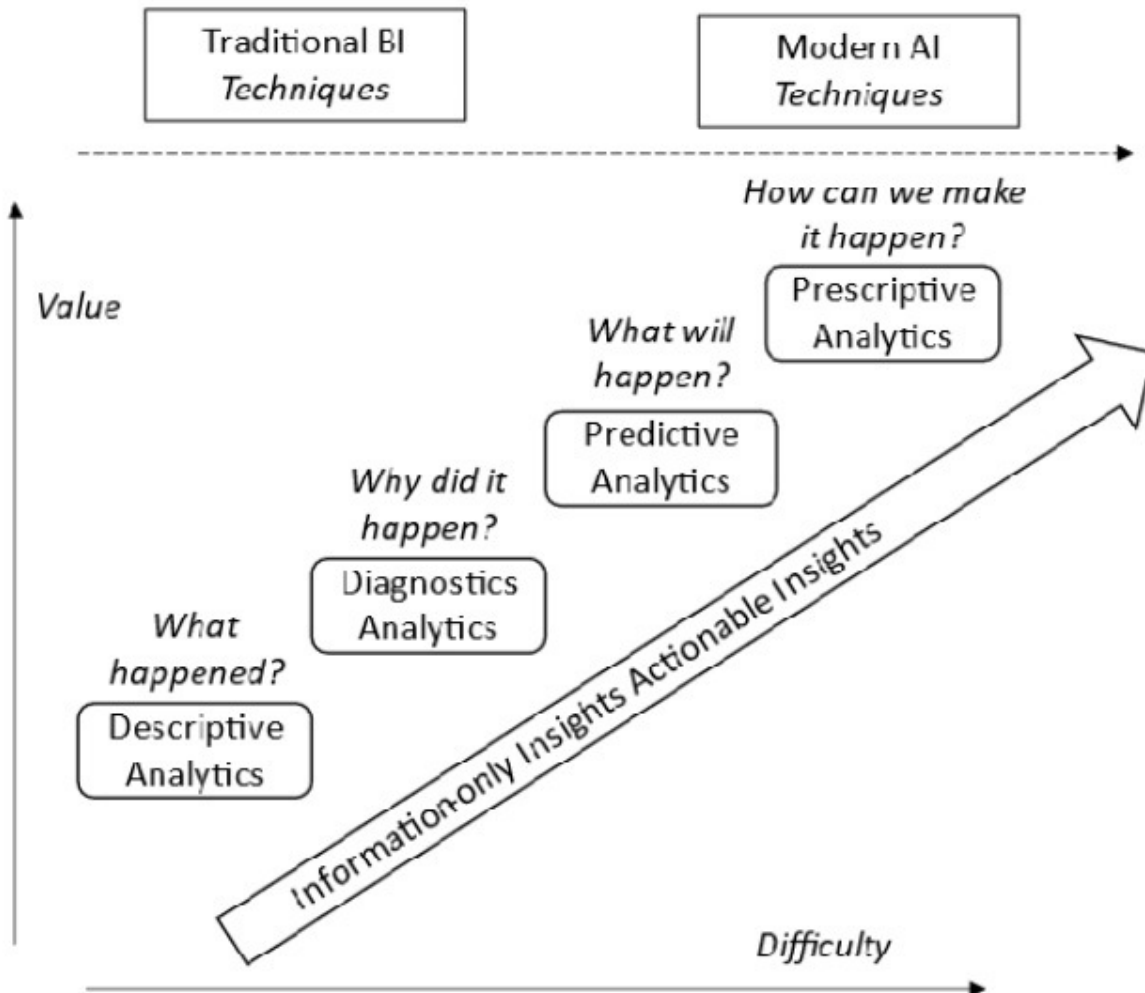


Figure 5.11 – Analytics techniques

Here, we can see that we have always been able to get insights from data, but there was always an *actions gap*. This gap has meant that while insights are great and can be very powerful, if they don't allow an organization to do something with that knowledge, people have just been *busy fools* analyzing data and presenting dashboards of charts without getting any value.

Only when we look at what modern techniques through AI can deliver, such as predictive analytics and self-actioning prescriptive analytics, can we start to see actual value creation from our data analysis work.

There is a very close relationship between BI and AI. BI uses analytics services, which can be considered a suite of business analytics tools for analyzing data and sharing insights, but what value do they provide? And what do we do with those insights?

This is where AI comes into the picture; AI refers to the ability to analyze large quantities of data, learn from the results, and then use this knowledge (*insights*) to optimize and change future processes, systems, and so on. It is often said that the most challenging part of AI isn't AI... *it's data!*

It is also said that you should not *AI before you BI*; that is, you should place visibility and transforming data ahead of data intelligence (*garbage in, garbage out*). BI is often the bridge between AI and the data. This approach can be visualized in the following diagram:

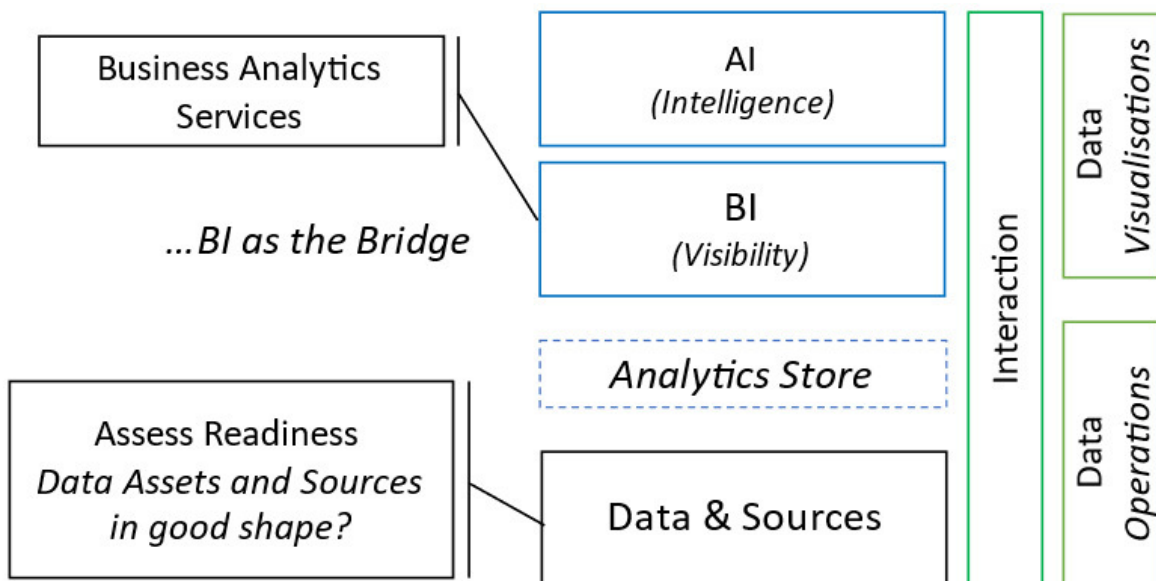


Figure 5.12 – Analytics as a bridge

The first stage in any big data or analytics project should be to assess the readiness of the data, ensuring that any data assets and sources are in good shape for analysis.

We refer to this as *preparing* the data.

The following are the characteristics you should consider for big data:

- **Quantity (volume) of data arriving (ingesting):** This is often on the petabytes scale.
- **Speed (velocity) of data arriving (ingesting):** This could be near/real-time streamed data, a time-framed schedule, or batch data.
- **Age (validity) of data arriving (ingesting):** This data could have a life cycle that means the data value is no longer valid. This is critical for decisions or actions based on a value.
- **Format (variety) of data arriving (ingesting):** This could be structured, semi-structured, or unstructured data.

Traditional database systems and data stores do not have the characteristics of the preceding list. To allow us to move beyond the limitations of the traditional DW and move toward more intelligent and actionable insights, we need a paradigm shift in terms of our mindset and technologies; we need a modern set of technologies and a new architecture is required to support this modern way of *ingesting, processing, analyzing, and visualizing* data to *take action*.

As we saw earlier in this section, *big data* is more than just the volume of data; when we combine the velocity of the data, its complexity, and the format (being unstructured), we get the term *big data*.

The traditional **ETL** toolset of **extracting (E)** the data from sources before **transforming (T)** and then **loading (L)** it into destinations cannot keep pace with the velocity and variation of this big data. Big data and the modern DW bring rise to this need to shift from ETL to **Extract, Load, Transform (ELT)**, which supports changing the volume of data and the nature of its complexity.

Much like the DevOps world brought *Development* and *Operations* teams closer together for the better, bigger goal, the big data era brings *data scientists, data engineers, database administrators, IT pros, and business analysts* closer together. This is the birth of *unified analytics*. The following diagram represents a big data and analytics architecture model:

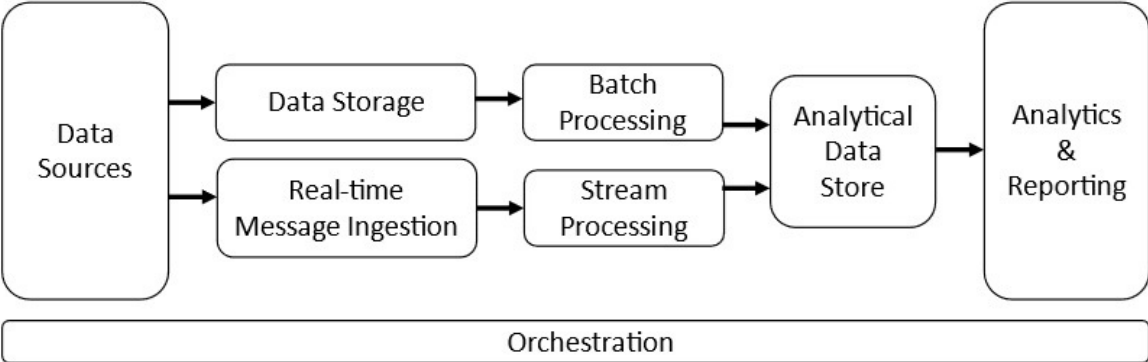


Figure 5.13 – Big data and analytics architecture

One or more of the components in the preceding diagram can be part of the big data architecture:

- **Data Sources:** Databases, log files, file stores, IoT devices, and social media
- **Data Storage:** Data lakes and blob storage

- **Real-Time Message Ingestion:** Azure Event Hubs, Azure IoT Hubs, and Kafka
- **Batch Processing:** HDInsight and Databricks
- **Stream Processing:** Azure Stream Analytics and HDInsight
- **Analytical Data Store:** Azure Synapse Analytics, SQL Data Warehouse, and HDInsight
- **Analysis and Reporting:** Azure Synapse Analytics, Azure Analysis Services, Power BI, and Excel
- **Orchestration:** Data Factory

There are several benefits of this architecture, such as the number of technology choices it provides and the ability to use open source (*Kafka*, *Spark*, *Hadoop*, and so on) as well as native Microsoft products and services. It also provides an elastic scale and parallelism, interoperability, and integration with existing enterprise solutions. However, there are some challenges, such as its complexity, skillset, the technology's maturity and its evolving nature, and security/privacy/compliance. The following diagram aims to position the Microsoft data landscape on Azure visually:

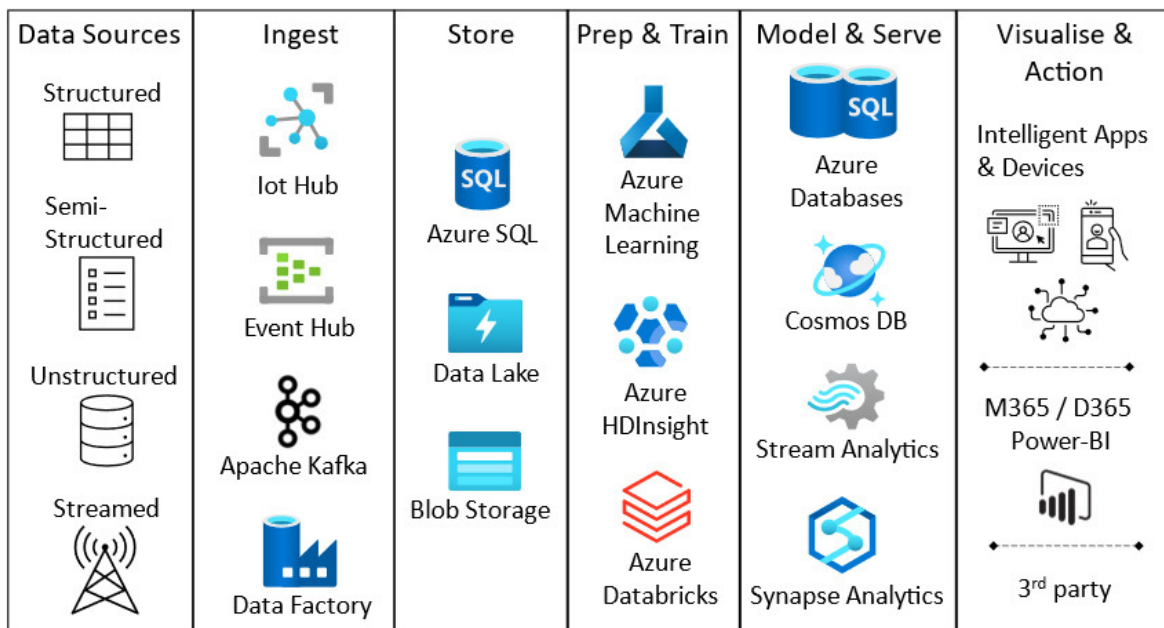


Figure 5.14 – Azure data landscape

The preceding diagram is by no means exhaustive but indicates the pillars in a data model architecture and where common services and solutions typically sit within that.

Many of the services and solutions that are shown in the preceding diagram are beyond the scope of this book; we have included some links in the *Further reading*

section of this chapter, should you wish to continue studying this area.

For the *exam objectives*, we will look at the following three Azure big data and analytics services:

- **Azure Synapse Analytics**
- **Azure HDInsight**
- **Azure Databricks**

This section looked at the data landscape. In the next section, we will look at Azure Synapse Analytics.

Azure Synapse Analytics

Azure Synapse Analytics is the rebranded cloud-based Azure SQL Data Warehouse. Other than just a name change, it has added functionality to provide capabilities that support big data; it now combines a modern Data Warehouse with a powerful and fully-featured scalable analytics service.

It is provided to users as a fully managed PaaS solution and is built on the *massively parallel processing (MPP)* relational database technology; it supports Apache Spark as a fully managed service, known as Spark-as-a-Service. Data is managed and interacted with through Azure Synapse Studio, which is a browser-based user interface.

Azure Synapse Analytics is intended for large-scale modern Data Warehouse and analytic scenarios with petabytes of data, with complex queries running against it.

Azure Synapse runs on clusters that contain the following components:

- Synapse SQL Engine
- Apache Spark integration service
- Data integration layer
- Azure Synapse Studio (*browser-based user interface*)

The following diagram outlines how Azure Synapse Analytics may be used in a solution architecture:

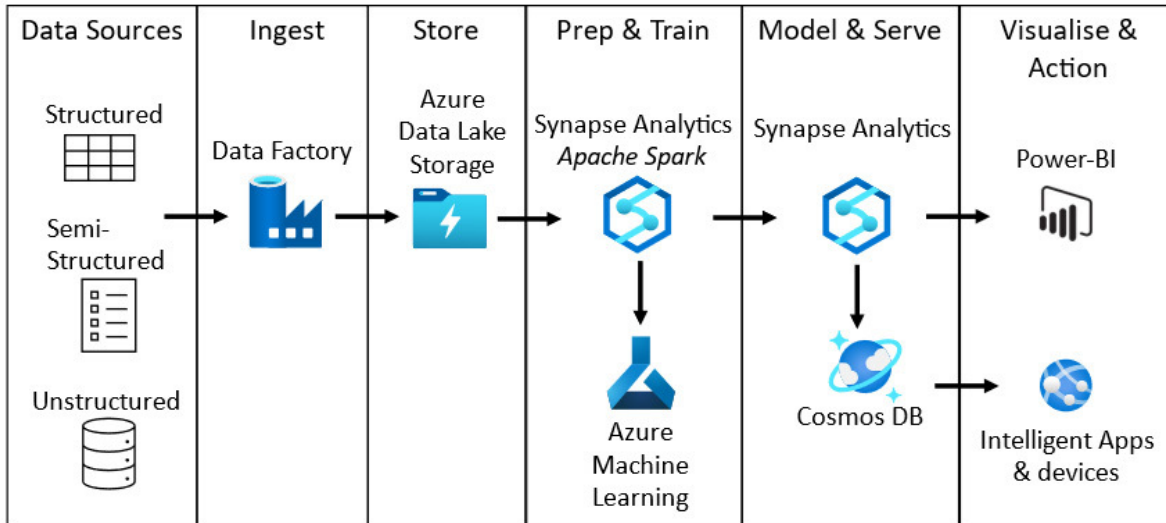


Figure 5.15 – Azure Synapse Analytics solution architecture

This section looked at Azure Synapse Analytics and how it can be used in a big data and analytics solution. The next section will look at how Azure HDInsight can be used in a big data and analytics solution.

Azure HDInsight

Azure HDInsight is a cloud distribution of Hadoop in Azure that's a fully managed analytics service for enterprise-scale organizations. It can be considered a scale-out data cluster compute engine that allows you to efficiently tackle complex unstructured data that can be ingested into a data store (*typically, a data lake*).

It uses the Hadoop open source framework for distributed processing and analyzing big datasets through clusters. Utilizing the advantages of cloud services makes this processing and analysis more accessible, faster, and more cost-effective. You can think of HDInsight as Hadoop-as-a-Service.

Although the clusters primarily run Hadoop as a managed service, there is also cluster support for the *HBase*, *Storm*, *Spark*, *Interactive Query*, *R Server*, and *Kafka* open source technologies.

The following are some examples of how HDInsight is being used in the real world:

- **Telecoms:** Churn prediction, market offers, pricing *call detail records (CDRs)*, network monitoring, demand provisioning, and optimizations.
- **Financial Services:** Customer 360 and fraud detection.
- **Health Care:** Clinical trial selection, patient mining, vaccine effectiveness, personalized medication, and health plans.
- **Industry:** Predictive maintenance, supply chain, stock control, and inventory optimization.
- **Utility:** Demand prediction and smart metering.

The following diagram outlines how Azure HDInsight may be used in a solution architecture:

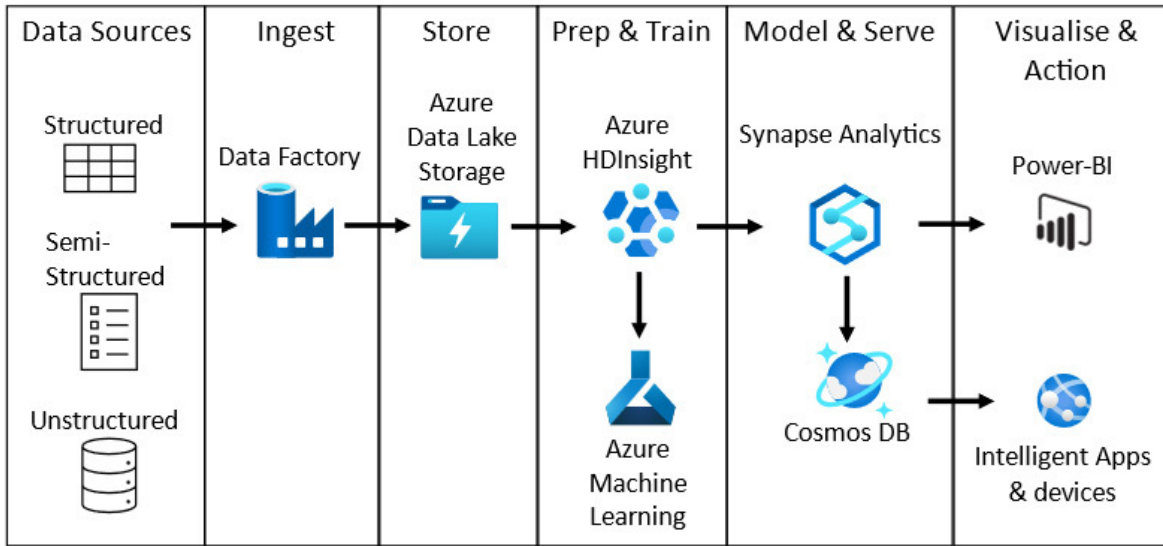


Figure 5.16 – Azure HDInsight solution architecture

This section looked at Azure HDInsight and its use in a big data and analytics solution. The next section will look at Azure Databricks and how it can be used in a big data and analytics solution.

Azure Databricks

You can think of **Azure Databricks** as *Spark-as-a-Service*. **Azure Databricks** is an *Apache Spark-based* big data analytics PaaS service; it provides fully managed Spark clusters that Databricks runs on top of. This means that there is no need to provision VMs or master and worker nodes; this requirement has been abstracted (*removed*) and is provided as the managed nodes service.

Data is managed and interacted with through the Databricks workspace, which is a browser-based user interface.

It can be used in machine learning solutions to prep and train large amounts of data through its optimal design, which was made for such scenarios. As such, it includes the *MLlib* machine learning library; it also has a use case for IoT solutions and streaming near/real-time data. The following diagram outlines how Azure Databricks may be used in a solution architecture:

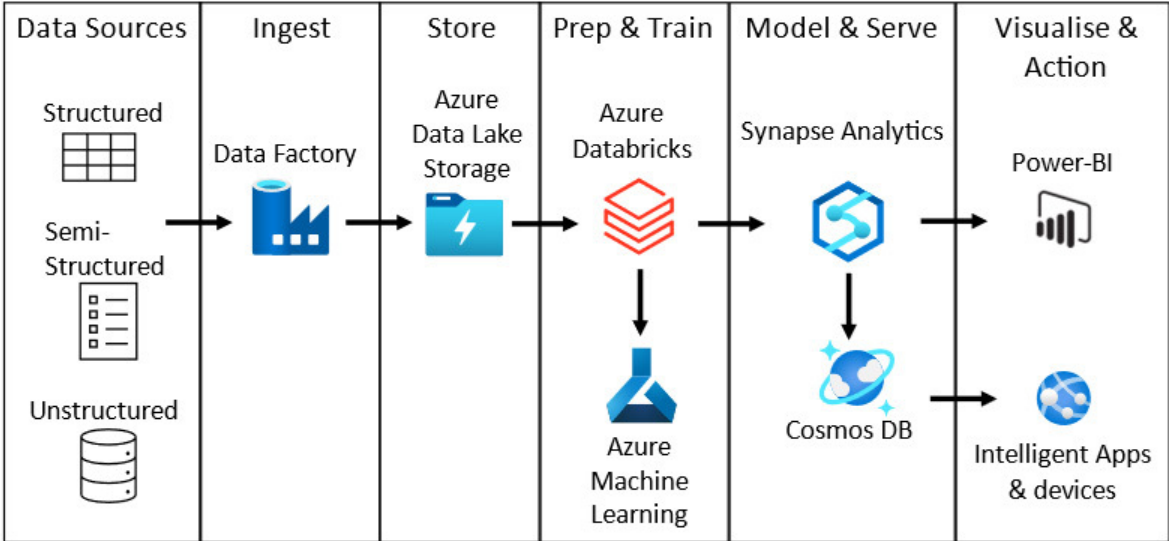


Figure 5.17 – Azure Databricks solution architecture

This section looked at Azure Databricks and how it can be used in a big data and analytics solution. The next section will look at Azure DevOps solutions.

DevOps solutions

This section will aim to cover the following exam objective: *describe the benefits and usage of Azure DevOps, GitHub, GitHub Actions, and Azure DevTest Labs.*

Before we look closer at the **DevOps solutions** that are part of the exam objectives, we will introduce DevOps and create a knowledge foundation and baseline to build from. This also aims to build an understanding of the bigger picture of how the different *DevOps solutions* came into existence, how they are positioned, and how they are interrelated for both *technical* and *business* personas.

The history of DevOps

A Tale of Two Teams

To understand *DevOps* as a software and systems development culture, it is important to understand its origins and why it came into existence.

The *waterfall* and *agile* software and systems development frameworks have their limitations; comparing them against DevOps is far beyond the scope of this book. For this book, you can consider waterfall and agile to be *development frameworks* and DevOps as a *development culture*.

We should recognize that agile's approach to development is closer to DevOps than waterfall's as they share many of the same goals. The approach of agile is in contrast to waterfall's; there is no *fixed scope*, only an *estimated scope*, with development consisting of many iterative changes and loops of the cycle, as shown in the following diagram. These development iterations will be working closely with the customer and stakeholders but will not involve the operations silo. The following diagram shows the fundamental differences between the *waterfall* and *agile* frameworks and how their approach differs to development:

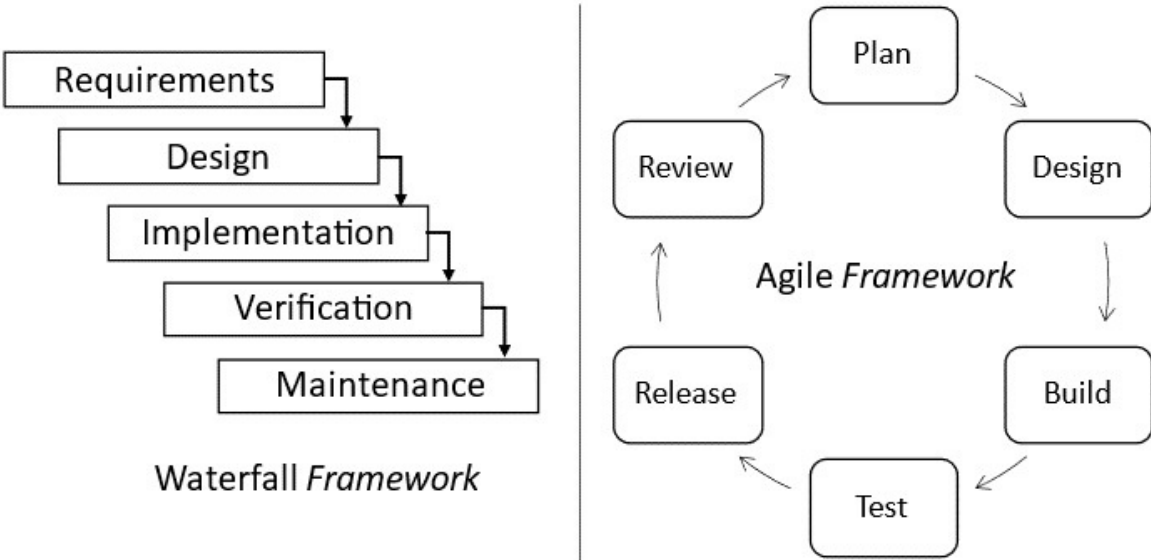


Figure 5.18 – Waterfall and agile frameworks

This development occurs in silo teams, where there is no collaboration between developers and *operations engineers*, which is at the root of the issue that *DevOps* sets out to address.

Between these two team silos, there has traditionally been a certain lack of understanding of each other's perspectives. With a lack of trust and communication, politics, no shared interest, and goals that are opposing, there is a melting pot of issues that do nothing to help the goal of building, releasing, deploying, and operating software and systems to make them the best that they can be.

These two teams push in different directions and function at different levels. Developers want to release systems and software at speed, but operations engineers need steady control; this leads to friction. The development vision is fundamentally at odds with the historically cumbersome, static, and technical debt of operations. The following diagram visualizes these opposing cultures:

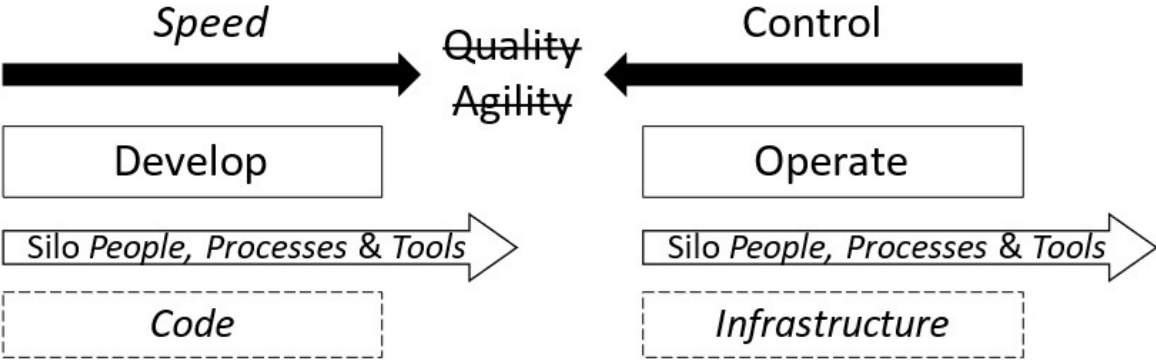


Figure 5.19 – Opposing team cultures

The developer's need for *speed* impacts *quality*, and the operation's need for *control* impacts *agility*.

These clashes between the *them* and *us* cultures and mindsets of development and operations were the key catalyst for the birth of DevOps.

This section looked at the origins of DevOps. In the next section, we will look at what DevOps is.

What is DevOps?

People | Process | Technology

DevOps is a *software and systems development culture* that's enabled through *team collaboration, common processes, and technology* to deliver a better aligned and unified outcome.

It was conceived as more of a *mindset* than a *toolset*; this means it's as much a cultural shift of silos of the **Development** and **Operations** teams as it is a technology. The hardest thing to change in any organization is not the **technology** but the **people** and **processes**.

Microsoft's definition of DevOps is that "*DevOps is the union of people, processes, and technology to enable continuous delivery of value to your end users.*"

Gartner's definition of DevOps is "*DevOps seeks to bridge the development and operations divide through the establishment of a culture of trust and shared interest among individuals in these previously siloed organizations.*"

The following diagram visualizes the DevOps process flow, which can help translate the concept of DevOps into tangible actions and value:

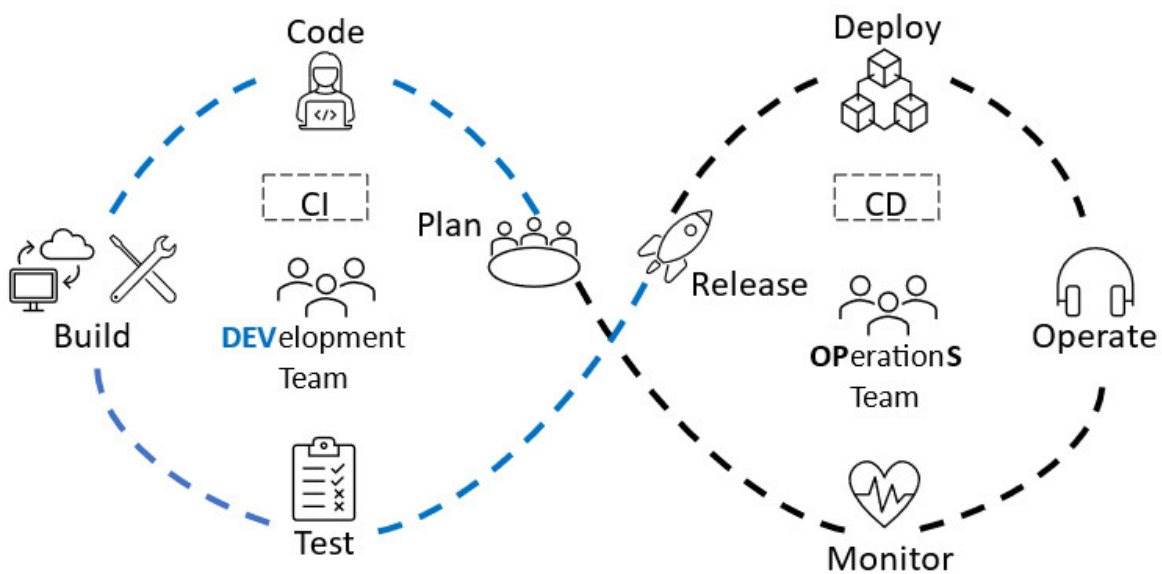


Figure 5.20 – The DevOps process flow

Development and operations teams can be more effective when collaborating on working toward a common set of aligned goals and automating repetitive tasks. This means systems and software can be created, updated, and deployed much quicker with increased quality, removing cost and adding value at every stage. The following diagram shows the nirvana of DevOps as a shift in software and systems development culture; the goal of DevOps is to make workload delivery vastly more efficient:

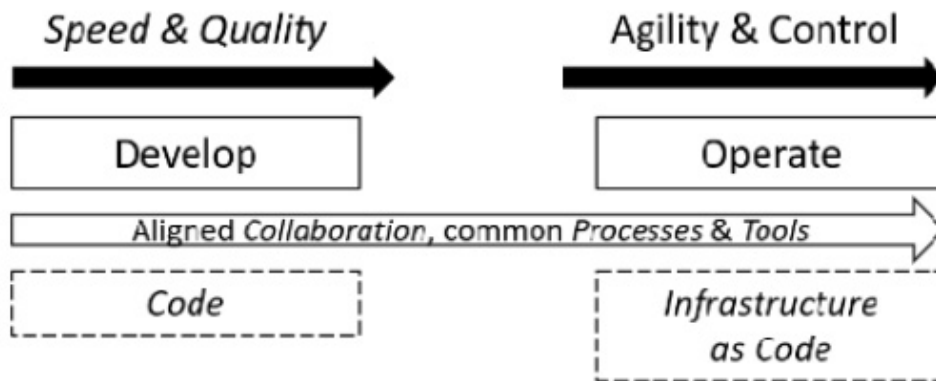


Figure 5.21 – Unified team culture and alignment of goals

Developers no longer trade quality for speed, and operations engineers adopt a more *agile* approach to deploying infrastructure that is balanced with *control*. This means that operations teams deploy and manage their infrastructure not by clicking through a portal or manually executing shell commands but through *code*; they adopt *practices* and *tools* used by development teams and treat their infrastructure as *source code*. This is a concept known as **Infrastructure as Code (IAC)**.

A DevOps approach can be achieved by automating each stage of the process as much as possible, as well as fostering collaboration between the **Development** and **Operations** teams to deliver unified outcomes through aligned common goals.

In this section, we introduced what DevOps is. In the next section, we will look at some further concepts surrounding DevOps.

CI/CD

CI/CD is an abbreviation for **Continuous integration/continuous delivery**.

In a nutshell, these are a set of practices and operating principles that form the basis of the DevOps culture. This is to allow Dev and Ops teams to work more collaboratively,

align team goals, and release software and systems with greater frequency, control, and quality.

CI/CD is a continuum; CI automates the process of building code and testing it, at which point CD takes over to automate the delivery to the required environments for consumption. The deployment is then operated and monitored, and changes can be fed back into the loop to have the code edit cycle start again.

The following diagram visualizes the CI/CD approach and how it maps to the DevOps process flow:

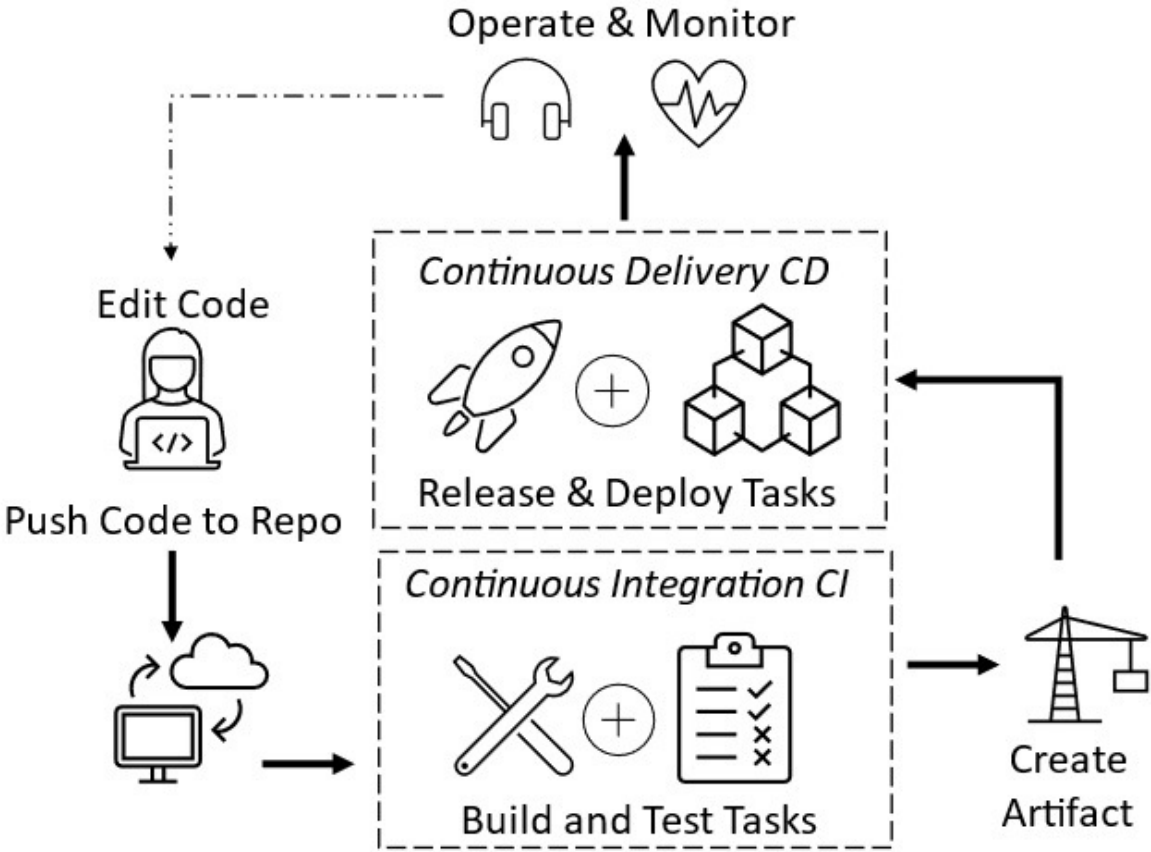


Figure 5.22 – Continuous integration/continuous delivery

As shown in the preceding diagram, CI/CD maps to the DevOps process flow that we learned about in the *What is DevOps?* section.

This section looked at the concept of CI/CD within DevOps. In the next section, we will look at the Azure DevOps service.

Azure DevOps

Azure DevOps (formerly **Visual Studio Team Services (VSTS)**) is an integrated *code development* and *deployment* platform.

Provided as a *Software-as-a-Service (SaaS)* platform by Microsoft, it can be accessed and used via the Azure portal; its purpose is to allow software and systems to be created while DevOps practices are being followed.

Azure DevOps is also available as an on-premises solution with *Azure DevOps Server*.

Azure DevOps provides tools and capabilities to manage complex software projects that require many individuals, often across many teams, and provides planning, tracking, software builds, versioning, testing, deployment, monitoring, and overall management of projects.

Azure DevOps is comprised of the following five services:

- **Azure Boards:** This is an agile project management and planning tool; it provides Kanban boards, tracking, reporting, and so on.
- **Azure Repos:** This is a centralized *repository* for source code that provides versioning tools, allows code reviews, and fosters collaboration.
- **Azure Test Plans:** This is a test management service.
- **Azure Artifacts:** This is a repository for hosting *artifacts* that can be pulled into *pipelines*; that is, compiled source code.

Azure Pipelines: This is a *continuous integration (CI)* and *continuous delivery (CD)* pipeline automation tool.

The following diagram visualizes the DevOps tools landscape:

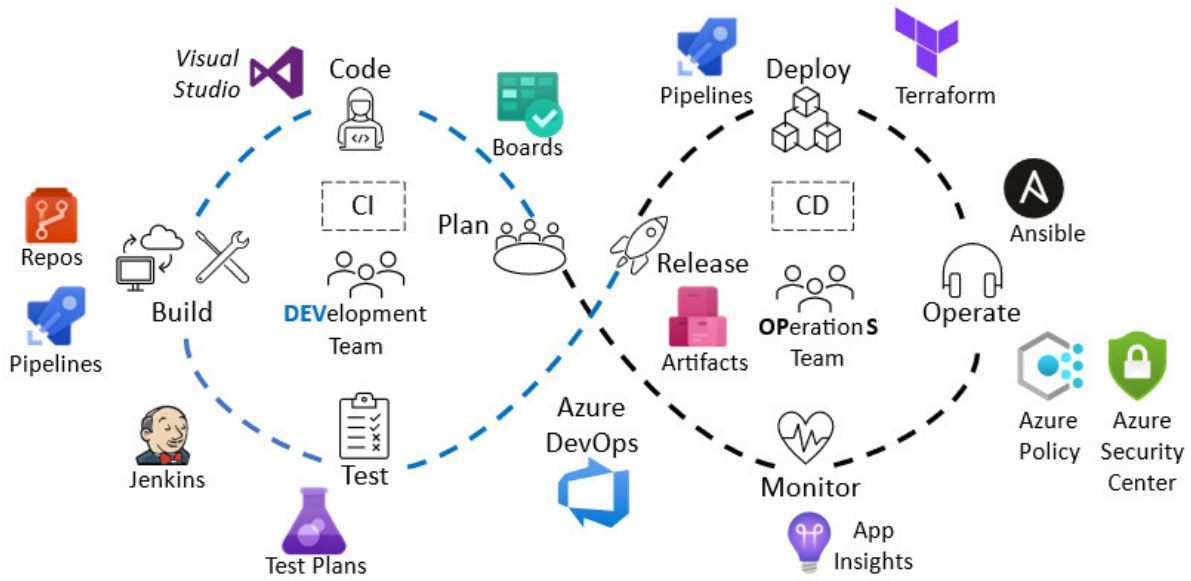


Figure 5.23 – Azure DevOps services

The preceding diagram outlines the related CI/CD aspects, the relevant Microsoft services that would be used, and third-party services. Third-party services such as *Trello*, *Jenkins*, *Terraform*, *Ansible*, *ELK Stack*, and so on can work in conjunction with Azure DevOps; you can use a *mix and match* approach to combine the best DevOps tools to meet your needs.

This section looked at Azure DevOps. The next section will look at GitHub and GitHub Actions.

GitHub and GitHub Actions

GitHub is a code hosting platform (*repository*) for open source software and is provided as an online hosted SaaS tool. Its equivalent Azure DevOps service is Azure DevOps Repos.

GitHub Actions is an API that allows you to automate/orchestrate CI/CD workflows.

GitHub provides more than just a code hosting platform; among other things, it also provides wiki-based collaborative communication, a knowledge base, and a documentation workspace.

Many of the features of DevOps tools overlap, so it can be hard to know when to choose which one. Azure DevOps is the choice for large enterprise teams (*although GitHub has its place also*) or those who require heavier project management, planning, control, and governance. You can mix and match using Azure DevOps Pipelines and Boards by using GitHub public code repositories.

This section looked at GitHub. The next section will look at Azure DevTest Labs.

Azure DevTest Labs

Azure DevTest Labs is an environment for automatically creating lab resources from pre-configured base items or ARM templates, all while utilizing the least administrative resources and effort.

This allows VMs and resources for testing and development purposes to be quickly created on-demand and then torn down. This can be done to integrate with a DevOps CI/CD approach and reduces the administrative effort needed to deploy a large number of machines, even if they have very different configurations and requirements and are required for a short duration, such as for build testing before releasing code.

Governance can be put in place to control how many labs can be created and how long they can run. The ability for teams to have these pre-configured base items, along with the necessary software and tools, means they can be far more productive and efficiently use resources as well as save costs. Once testing has been completed, resources can be shut down and de-provisioned.

In the next section, we will look at a thought exercise to reinforce what you've learned so far and improve your skills.

Thought exercise

In this exercise, we will look at a fictitious company called *MilesBetter Pizzas*.

With the *walk-in* trade no longer a viable commercial business model due to the pandemic, they have taken their business online through a pizza delivery service.

They wish to streamline operations, be as agile as possible, and react quicker than their competition to their customer's demands during the pandemic; solutions will need to allow them to react to the market, innovate and release online services, and provide initiatives quicker than their competition in these challenging times.

The approach to be taken should be based on understanding the business process requirements that a new app would be required to execute, the logic to be processed, and the outcome to be arrived at, and then translating that into the most appropriate technology choice, based on the decision's criteria.

The following diagram visualizes the core design paths:

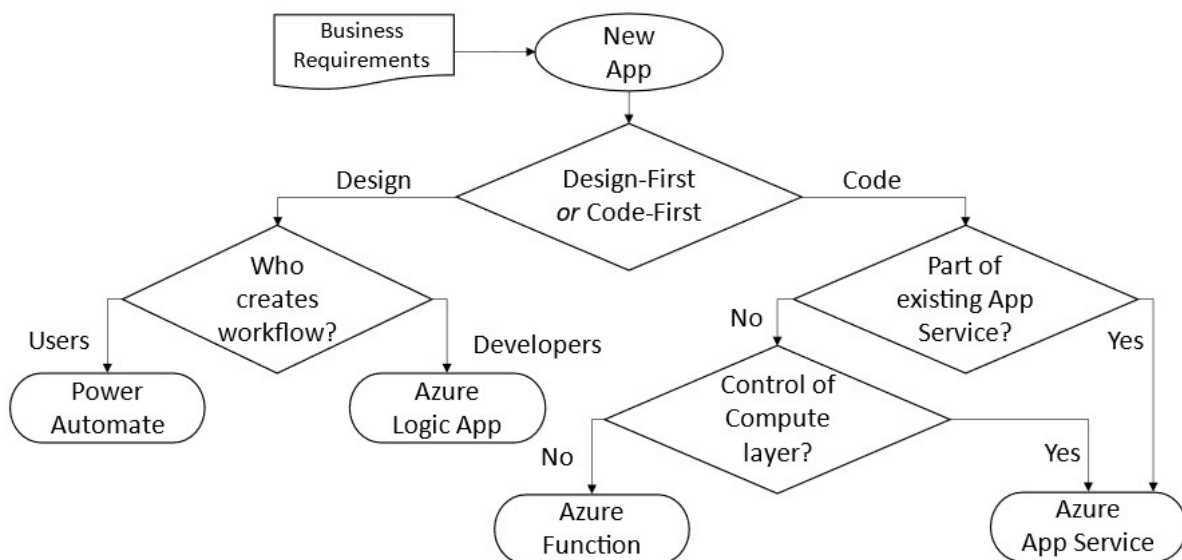


Figure 5.24 – Technology choice criteria

There are two core technology paths, *design-first technologies* and *code-first technologies*. The decision criteria are based on whether the workflows (*business processes*) will be written (and executed) in *code* or via a *graphical interface designer tool*. This will also be dictated by the resources available to design the workflow; are

there business analyst-type users with no coding experience or developers who will only work with code and coding toolchains?

The following section contains some hands-on exercises to reinforce the knowledge you've learned in this chapter and increase your skills.

Hands-on exercises

To support your learning with some practical skills, we will complete some hands-on exercises surrounding some of the resources covered in this chapter.

The following exercises will be covered:

- Exercise 1 – Creating a serverless solution using an Azure Function
- Exercise 2 – Creating a serverless solution using an Azure Logic App
- Exercise 3 – Creating an IoT solution using an Azure IoT Hub
- Exercise 4 – Creating an AI solution using a Bot Service

Getting started

To get started with these hands-on exercises, you must create a free Azure account at <https://azure.microsoft.com/free>.

This free Azure account provides the following:

- 12 months of free services
- \$200 credit to explore Azure for 30 days
- 25+ services that are always free

In addition, you will also need a Twitter account and an Outlook account to send notifications.

Exercise 1 – Creating a serverless solution using an Azure Function

This section will look at the steps to create a simple *Hello World* type of serverless solution example that will use a function based on an HTTP trigger. When we pass a value, code will be triggered to execute and return a response.

This could be the basis for a very simple inventory or assets, a lookup solution, or any solution that requires a message to be displayed when there is an HTTP request.

In the following subsections, the process of creating a solution has been segregated into tasks for ease of understanding.

Task – Accessing the Azure portal

1. Log into the Azure portal: <https://portal.azure.com>.

Task – Creating a Function App

2. In the search bar, type `function app`; click on **Function App** from the results list.
3. From the **Function App** blade, click on the **+ Create** button on the top toolbar.
4. From the **Basics** tab, set the **Project Details** settings as required.
5. Set the **Instance Details** settings as follows:
 - **Function App name**: Enter a name; this must be globally unique.
 - **Publish**: Leave this set to **code**.
 - **Runtime Stack**: Set this to **.NET**.
 - **Version**: Set this to **3.1**.
 - **Region**: Set this to the region that's closest to you.
6. Leave all other tabs with their default settings.
7. Click **Review + create**.
8. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make edits if required. Once you have confirmed your settings, click **Create**.
9. When the deployment is complete, you will receive a notification stating that the deployment succeeded. Now, click **Go to resource** from the **deployment** blade or navigate to the **Azure Function App** instance.

Task – Creating an HTTP triggered function

10. On the **Function App** blade, click the created **Function App**.
11. On the left menu, in the **Functions** section, click **Functions**.
12. Click **+ Add** from the top toolbar.
13. From the **Add function** pop-up window, click **HTTP trigger** from the **Select a template** section and click **Add**:

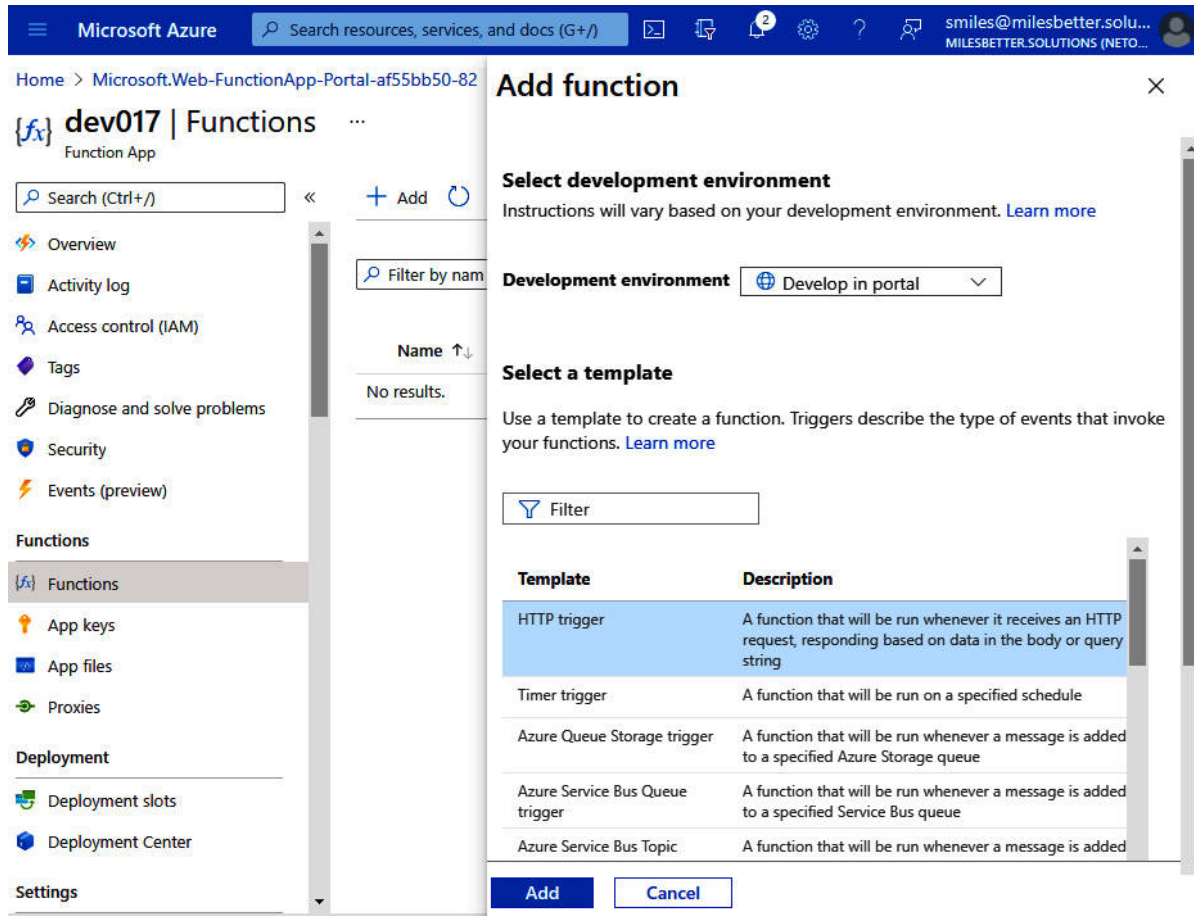


Figure 5.25 – Add function

14. From the **Function** blade, click **Code + Test** from the left menu:

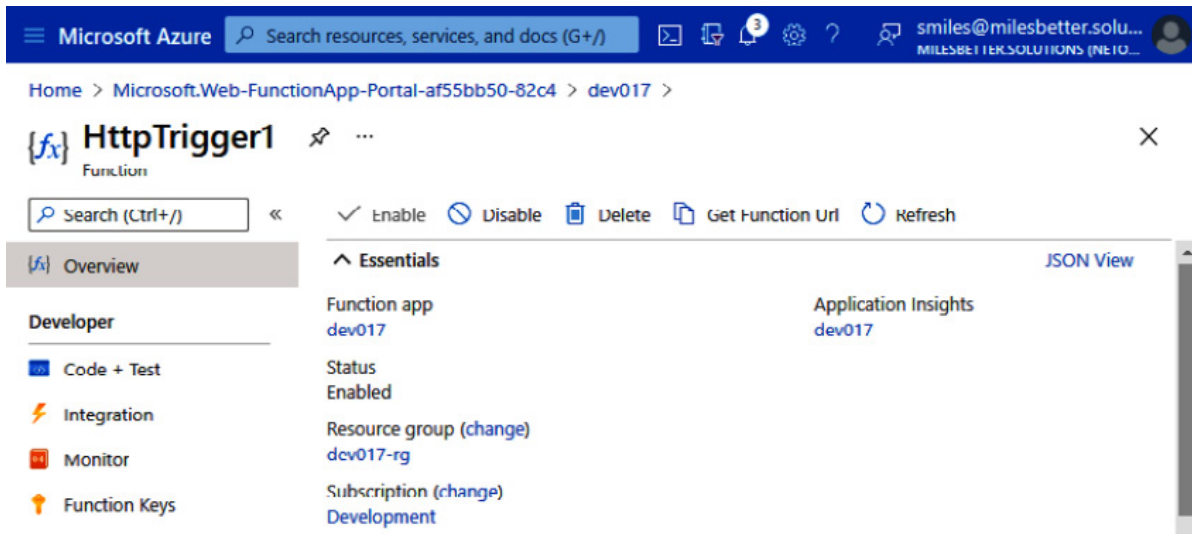


Figure 5.26 – Function blade

- From the **Code + Test** blade, you will see some auto-generated sample code; this will return a **hello (name)** message:

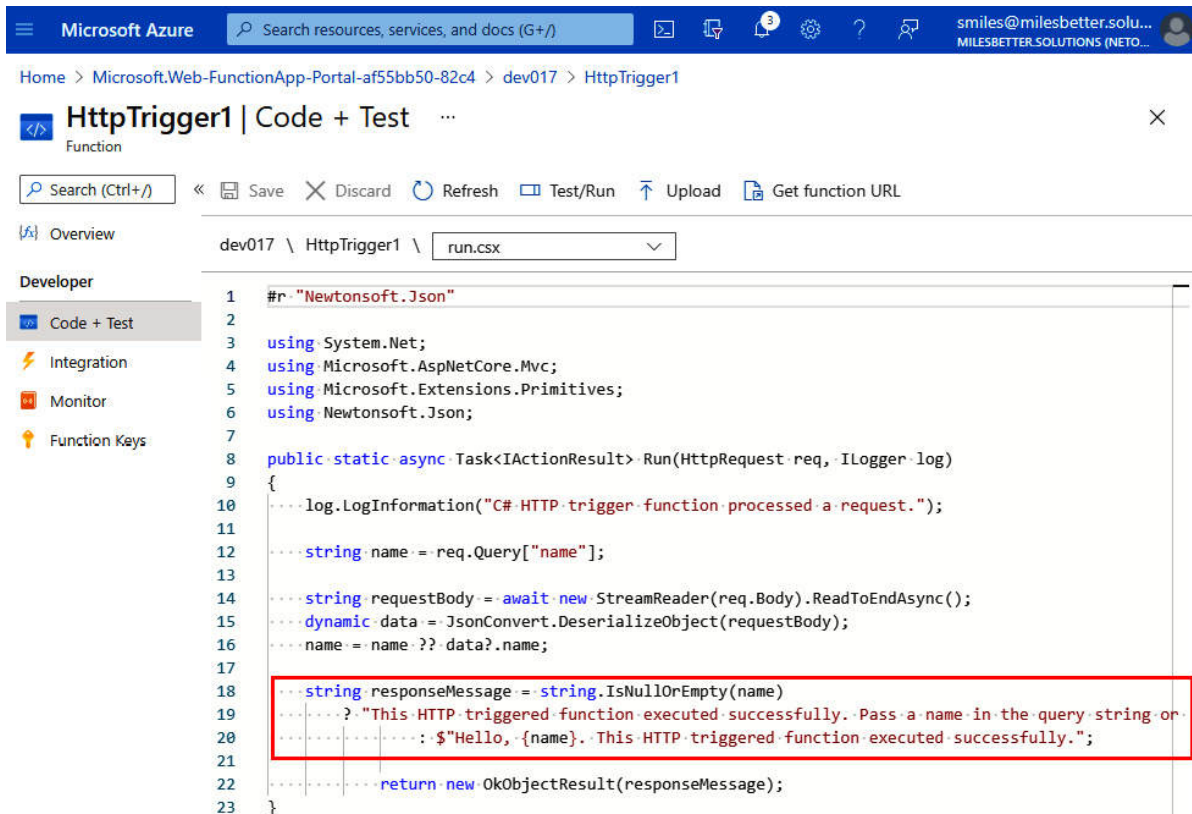


Figure 5.27 – Function app code

Task – Testing the function trigger

16. Click **Get function URL** from the top toolbar of the code editor and copy the function URL; the key value should be set to the default.
17. Open a new browser tab and paste in the copied function URL. The function will run when the page is requested and execute the code; this will prompt/require a name to be returned in the string or body response:

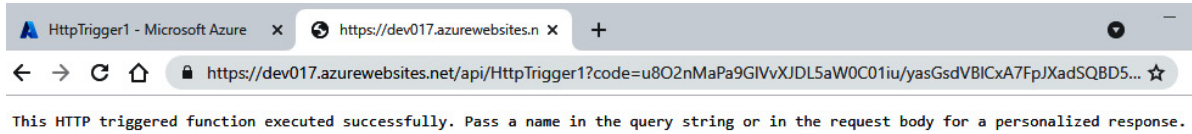


Figure 5.28 – HTTP URL trigger

18. Append **&name=yourname** to the end of the URL.
19. When you hit *Enter*, your function will run and return the name you entered:



Figure 5.29 – HTTP URL response

In this exercise, we looked at creating a Function App to display a `hello (name)` message when there is an HTTP request; that is, a URL is called. In the following exercise, we will look at creating a serverless solution using an Azure Logic App.

Exercise 2 – Creating a serverless solution using an Azure Logic App

This section will look at the steps to create an Azure Logic App that acts as a social media tracker; when a new tweet is posted that matches the set criteria, an email notification will be sent. Alternatively, it could notify a Teams or Slack channel, for example. This could be the basis for other solutions, such as an RSS feed tracker.

In the following subsections, the process of creating a solution has been segregated into tasks for ease of understanding.

Task – Accessing the Azure portal

1. Log into the Azure portal: <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – Creating a Logic App

2. In the search bar, type **logic apps**; click on **Logic Apps** from the results list.
3. From the **Logic Apps** blade, click on the **+ Add** button on the top toolbar, and then **+ Consumption**.
4. From the **Basics** tab, set the **Project Details** settings as required.
5. Set the **Instance details** settings as follows:
 1. **Logic App name**: Enter a name; this must be globally unique.
 2. **Region**: Set this to the region that's closest to you.
6. Leave all other tabs with their default settings.
7. Click **Review + create**.
8. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make edits if required. Once you have confirmed your settings, click **Create**.
9. When the deployment is complete, you will receive a notification that the deployment succeeded. Now, click on **Go to resource** from the **Deployment** blade or navigate to the **Azure logic app** instance.

Task – Creating a Twitter trigger

10. From the **Logic App** blade, click the created **Logic App**.
11. From the left menu, in the **Development tools** section, click **Logic app designer**.

12. From the **Logic Apps Designer** blade, scroll down to the **Templates** section and find the template named **Email yourself new tweets about a certain keyword via Outlook**:

TIP

You may want to pause for a second as you are scrolling through the templates to look at what else you can create – something that you may want to come back to later to try out. This may include the template that lets you share your tweets to Facebook, posts your Instagram posts to Twitter, notifies you when a file is created in Dropbox, copies a file to OneDrive, and so on.

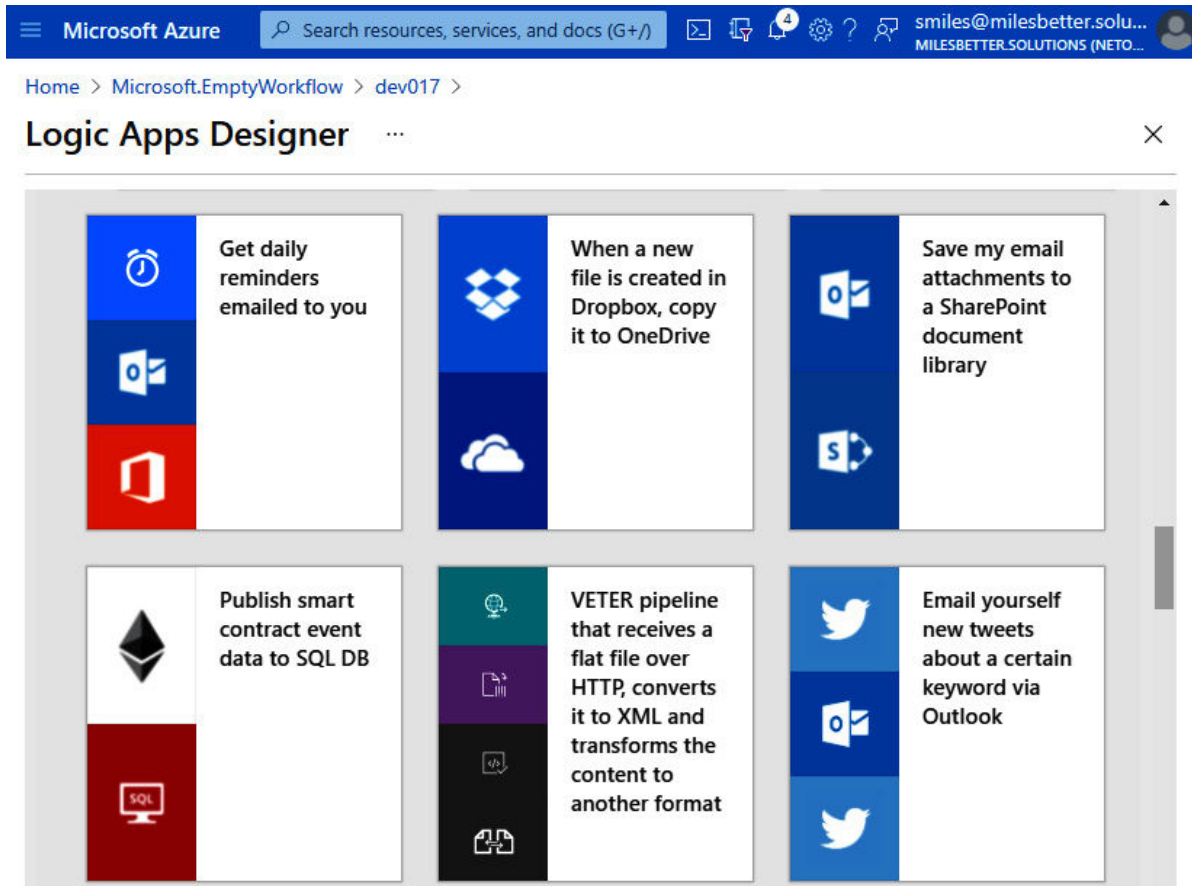


Figure 5.30 – Logic Apps Designer

13. Click on the template to open it and click **Use this template**.

14. From the **Logic Apps Designer** window, you must connect both your Twitter account and your Outlook account:

- **Twitter:** Click on the + symbol on the Twitter entry; enter a name for the connection, click **Sign in**, and follow the prompts to complete the sign-in process.
- **Outlook:** Click on the + symbol on the Outlook entry; enter a name for the connection, click **Sign in**, and follow the prompts to complete the sign-in process:

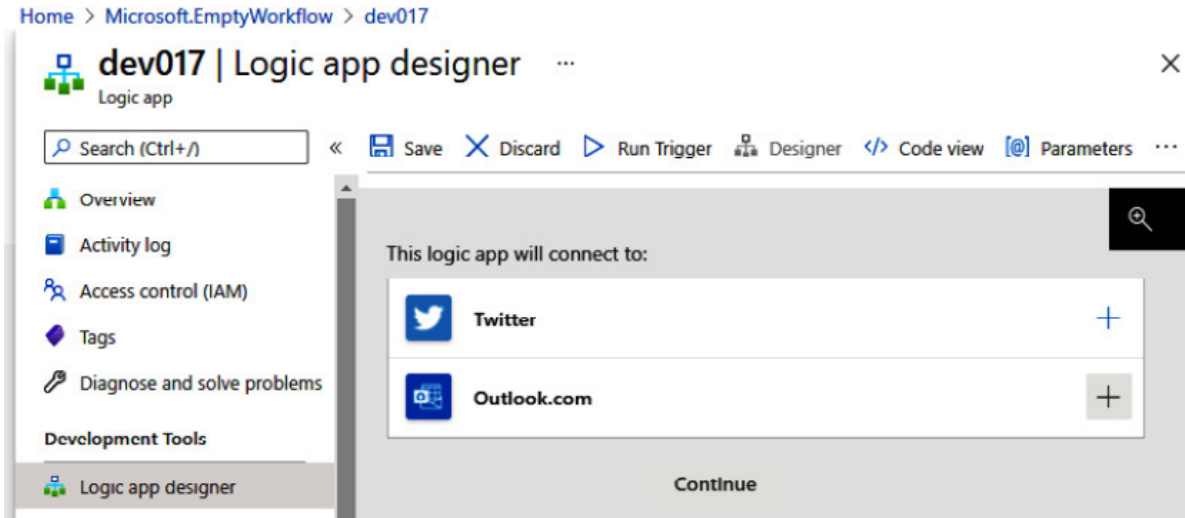


Figure 5.31 – Logic app connections

15. Click **Continue** once both Twitter and Outlook have successful connections; each item will have a green tick next to it.
16. In the **When a new tweet appears** box, enter the search text you wish to use; we will use **#AZ900FundamentalsPacktLogicAppTwitterDemo** in this example:

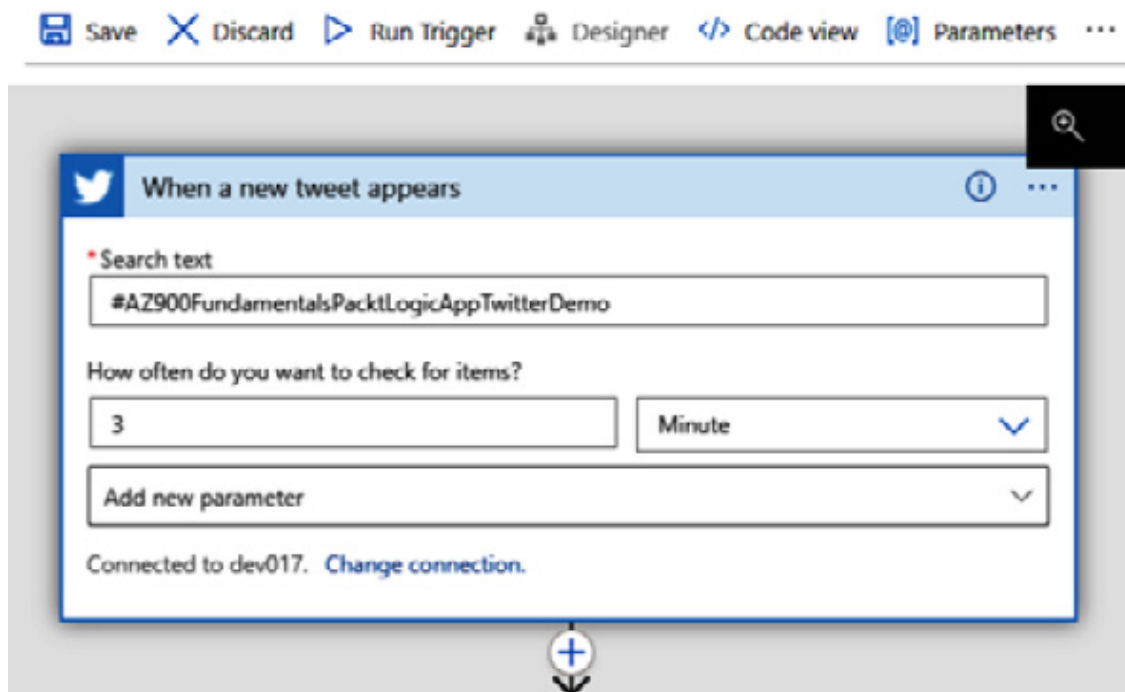


Figure 5.32 – Logic app workflow

17. In the **Send email** box, enter an email address that you want the notifications to be sent to:

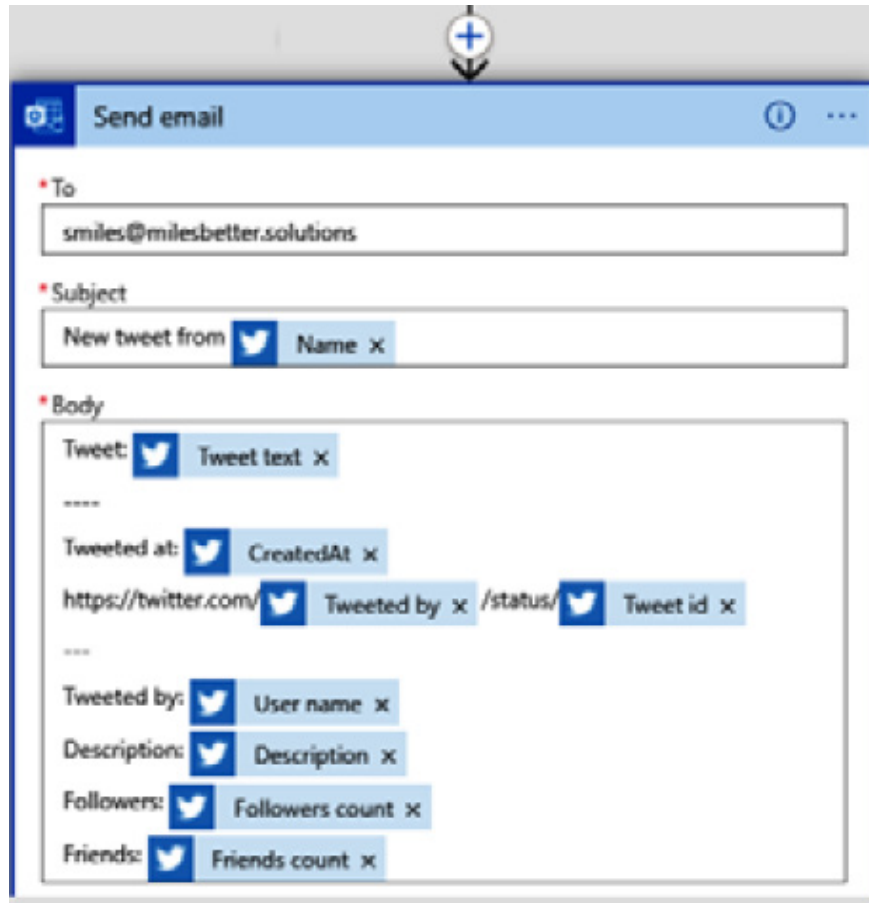


Figure 5.33 – Logic app workflow

18. Finally, click **Save** from the top toolbar. With that, you are ready to test.

Task – Testing the Twitter trigger

19. Click **Run Trigger** and then **Run** from the top toolbar:

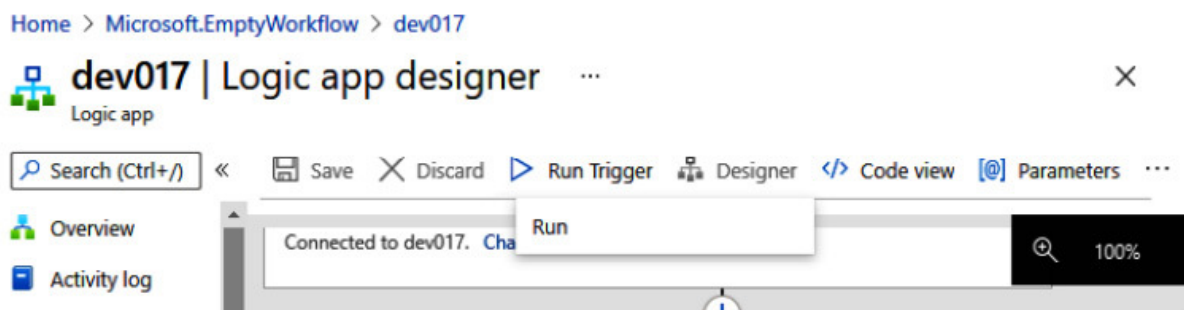


Figure 5.34 – Logic app Run Trigger

20. Open Twitter and paste in the search text we entered previously; that is, **#AZ900FundamentalsPacktLogicAppTwitterDemo**:

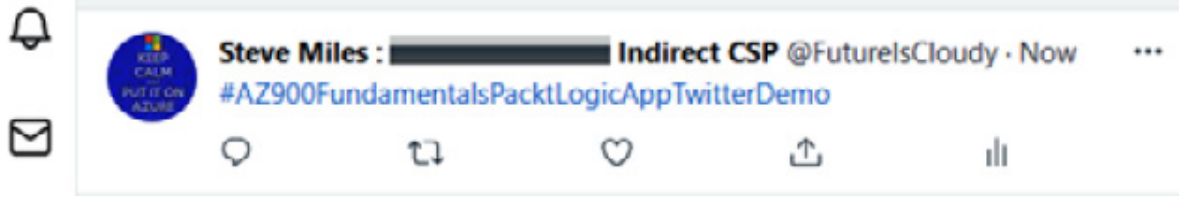


Figure 5.35 – Twitter post

21. You will receive an email notification of the new tweet:

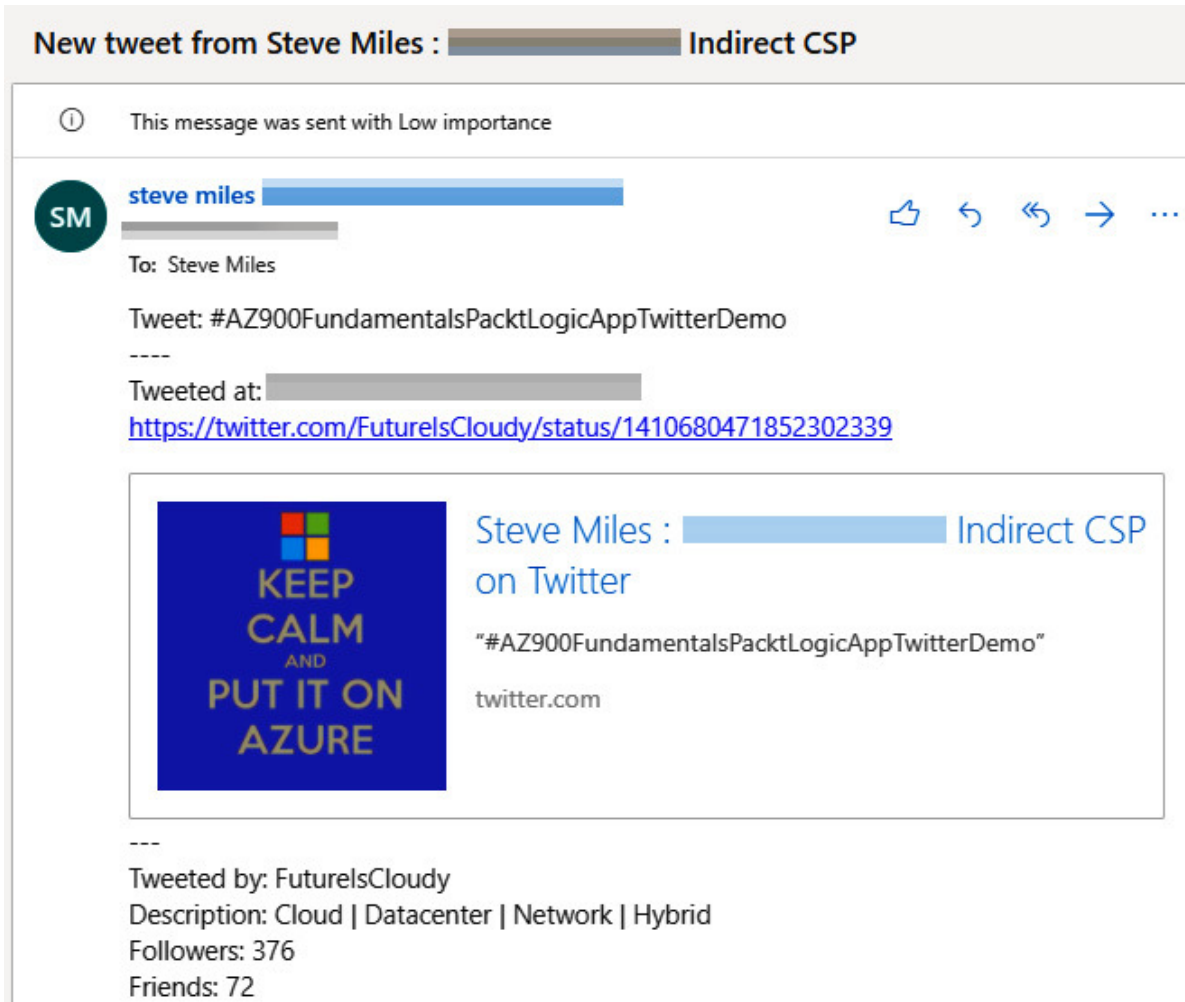


Figure 5.36 – Logic app email notification of tweet

In this exercise, we looked at creating a Logic App to send an email notification when there is a new Tweet containing a search term. In the following exercise, we will look at creating an IoT solution using an Azure IoT Hub.

Exercise 3 – Creating an IoT solution using an Azure IoT Hub

This section will look at the steps to create an IoT hub that can receive sensor data from a Raspberry Pi (*Sensor-Enabled Device*) simulator.

This could be the basis for any simple data collection solution from a sensor on any device.

In the following subsections, the process of creating a solution has been segregated into tasks for ease of understanding.

Task – Accessing the Azure portal

1. Log into the Azure portal: <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.



Task – Creating an IoT hub



2. In the search bar, type **IoT Hub**; click on **IoT Hub** from the results list.
3. From the **IoT Hub** blade, click on the **+ Create** button on the top toolbar.
4. From the **Basics** tab, set the **Subscription** and **Resource group** settings as required.
5. Set the region closest to you.
6. Enter a *unique* name for your IoT hub.
7. Click **Next: Networking**; leave the default settings as-is.
8. Click **Next: Management**; set **Pricing** and set **Scale tier** to **F1: Free tier**.
9. Click **Review + create**.
10. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
11. When the deployment is complete, you will receive a notification stating that the deployment succeeded. Now, click on **Go to resource** from the **Deployment** blade or navigate to the **Azure IoT Hub** instance.


Task – Creating an IoT device


12. From the **IoT Hub** blade, go to the **Explorers** section from the left menu and click **IoT devices**.
13. From the top toolbar, click **+ New**.
14. Provide an ID for your device and click **Save**:


Create a device ... ✕


 Find Certified for Azure IoT devices in the Device Catalog 


Device ID * 
 


Authentication type 
Symmetric key X.509 Self-Signed X.509 CA Signed

Primary key 

Secondary key 

Auto-generate keys 

Connect this device to an IoT hub 
Enable Disable

Parent device 
No parent device
[Set a parent device](#)

Save

Figure 5.37 – Create a device

15. Click **refresh** from the top toolbar if your device does not appear. Then, click your device to open its respective blade.
16. From the **Device** blade, copy the **Primary Connection String** value:

TIP

This will be used in the next task to authenticate to the Raspberry Pi simulator.

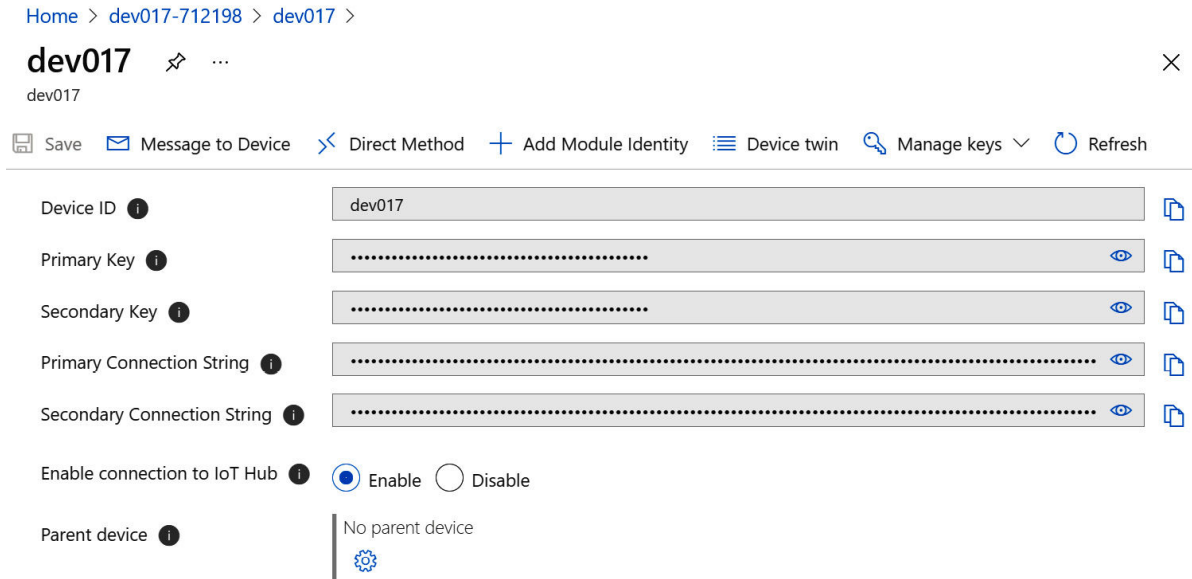


Figure 5.38 – IoT device connection string

Task – Configuring the Raspberry Pi simulator

17. From a browser, enter the following URL: <https://aka.ms/RaspPi>.
18. Once the simulator has loaded, click on the **Next** button shown as *Step 1* on the **Simulator Wizard** screen.
19. As shown on the **Simulator Wizard** screen, as *Step 2*, review the information shown regarding copying the device connection string from the Azure portal IoT hub you created in the previous task. Then, click the **Next** button.
20. As shown on the **Simulator Wizard** screen, as *Step 3*, review the information shown regarding placing the placeholder on line 15 with the Azure IoT Hub device connection string, as well as the instruction to click **Run** or type `npm start` in the console window to run the application. Then, click the **Got it** button.
21. From the top-right code area, locate line 15, which shows `const connectionString =`.
22. Replace the following entry with the Azure IoT Hub primary key connection string you copied in the previous task; that is, `'[Your IoT hub device connection string]'`:

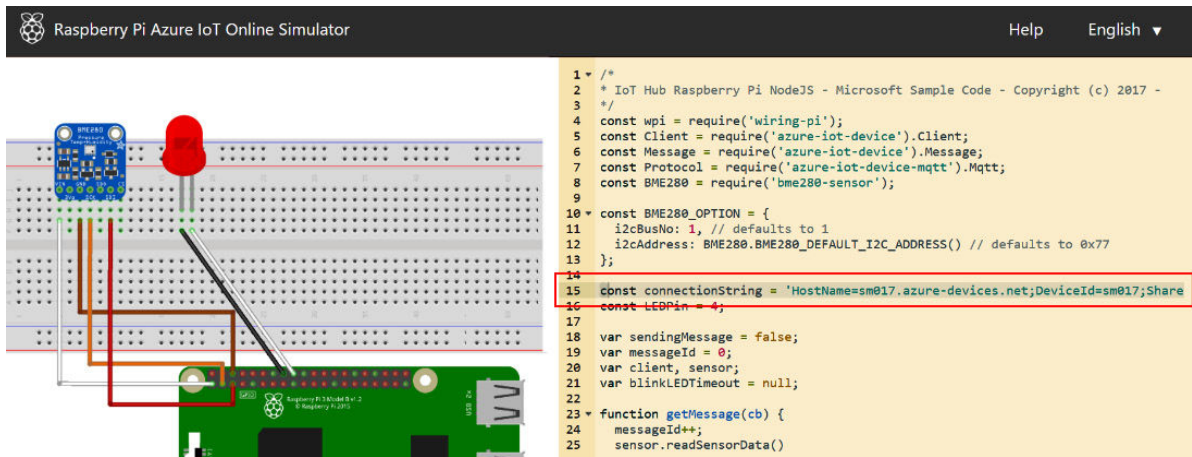


Figure 5.39 – Raspberry Pi simulator

Task – Testing the solution

23. Click **Run** or type `npm start` to run the application from the bottom-right console area.
24. You will now see messages stating that data is being sent from the device to the Azure IoT Hub; the red LED will also flash:

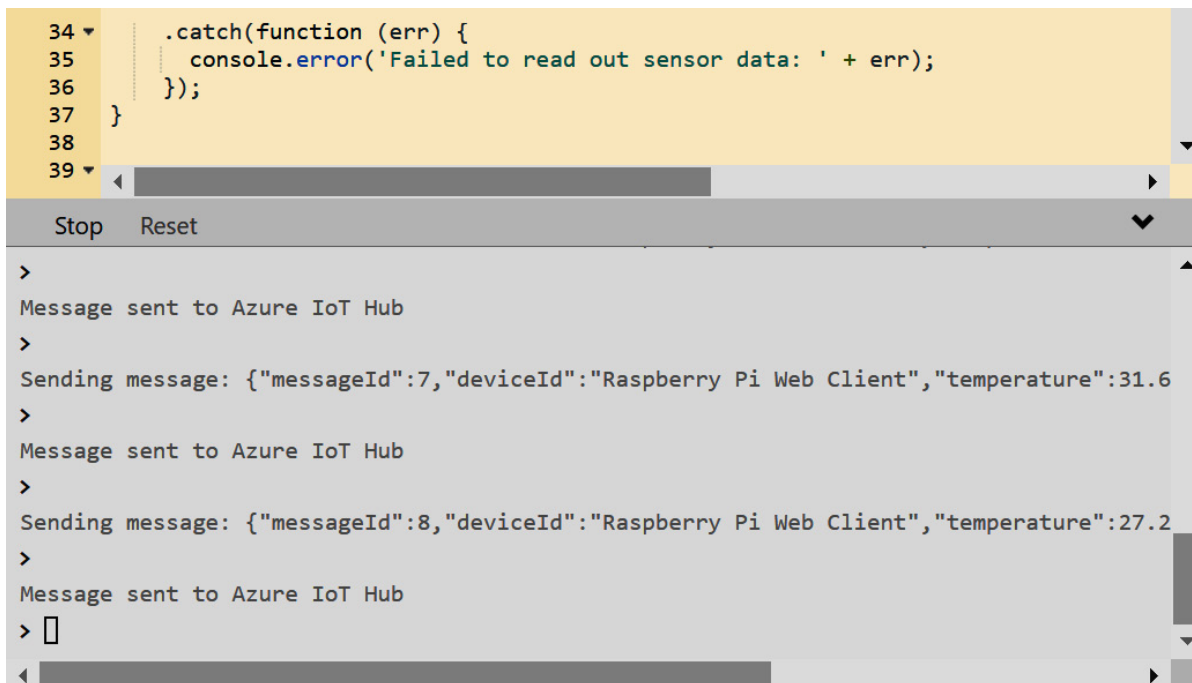


Figure 5.40 – Raspberry Pi simulator

25. Click **Stop**; you will see a message stating that the sample has stopped. The red LED will also stop flashing.
26. From the Azure portal, access the **Azure IoT Hub** blade. Then, from the **Overview** section, scroll down to view the messages that have been received:

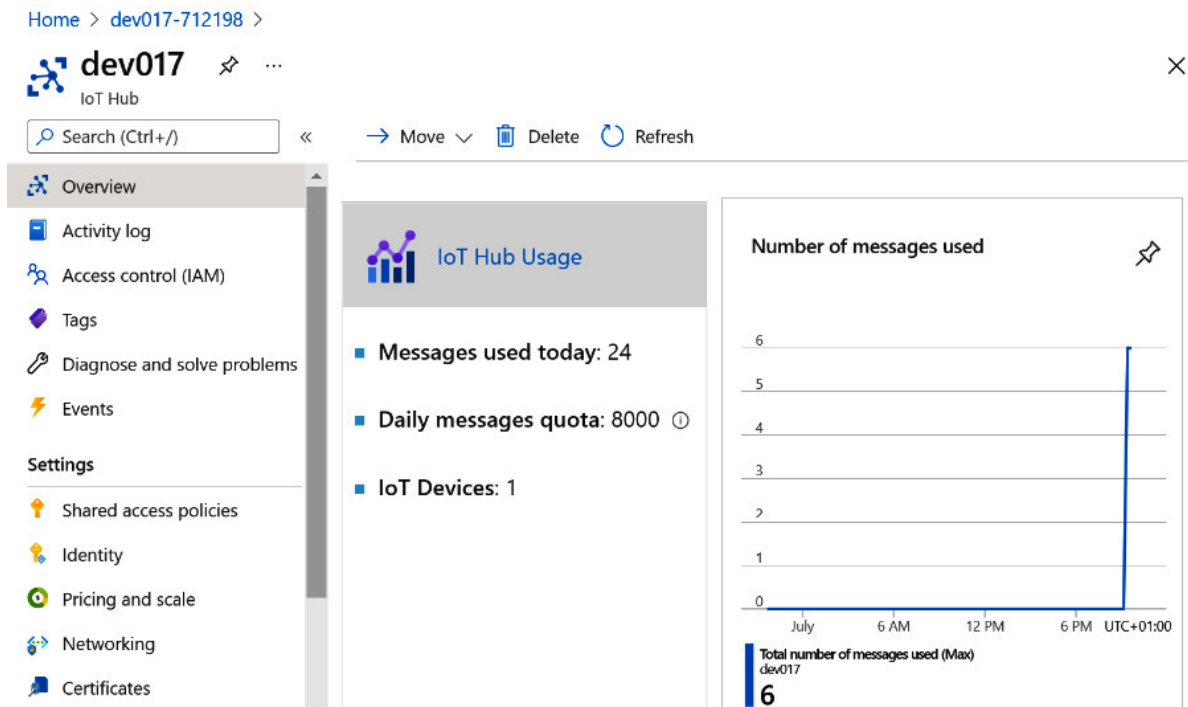


Figure 5.41 – IoT Hub Usage

In this exercise, we looked at creating an Azure IoT hub and sent it messages from a device with a sensor using a simulator. In the following exercise, we will look at creating an IoT solution using a Bot Service.

Exercise 4 – Creating an AI solution using a Bot Service

This section will look at the steps to create a Bad Joke *Knowledge Base (KB)* Bot Service as part of AI Cognitive Services. This could be the basis for building any customer services chatbot or conversational client application.

In the following subsections, the process of creating a solution has been segregated into tasks for ease of understanding.

Task – Accessing the Azure portal

1. Log into the Azure portal: <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – Creating a QnA maker

2. From a browser, enter the following URL: <https://www.qnamaker.ai>.
3. Sign into this site using the Microsoft account associated with the Azure subscription you wish to use.
4. Once you've signed in, from the **Create a knowledge base** screen, click on **Create a QnA Service** via **STEP 1** of the **QnA Maker** site. Upon doing this, you will be redirected to the Azure portal, which will open in a new browser tab:

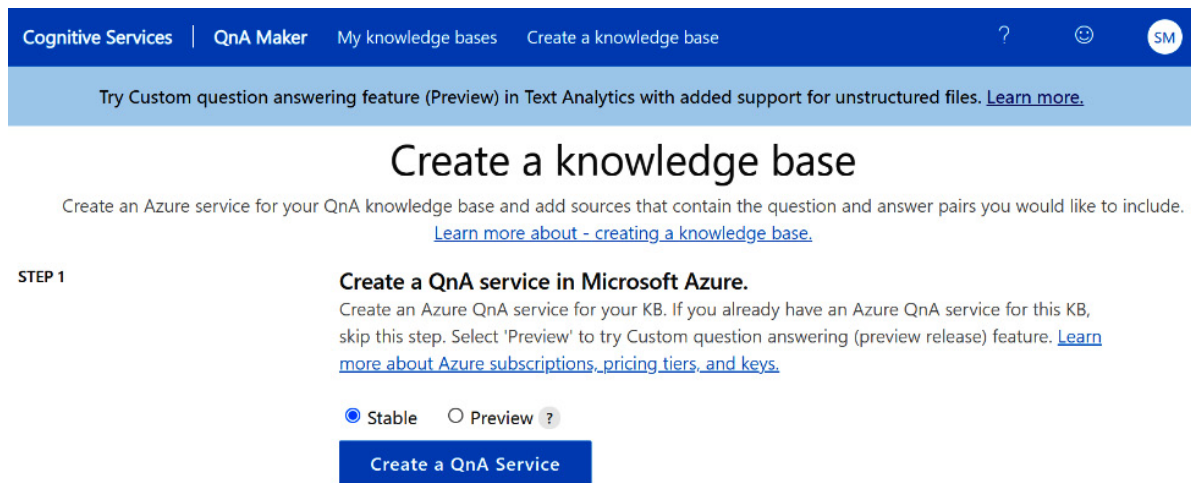


Figure 5.42 – Create a QnA Service

5. From the **QnA Maker Create** blade, from the **Basics** tab, set the **Subscription** and **Resource group** settings as required:
 - **Name:** Enter a unique name to be used.
 - **Pricing tier:** Select **Free F0**.

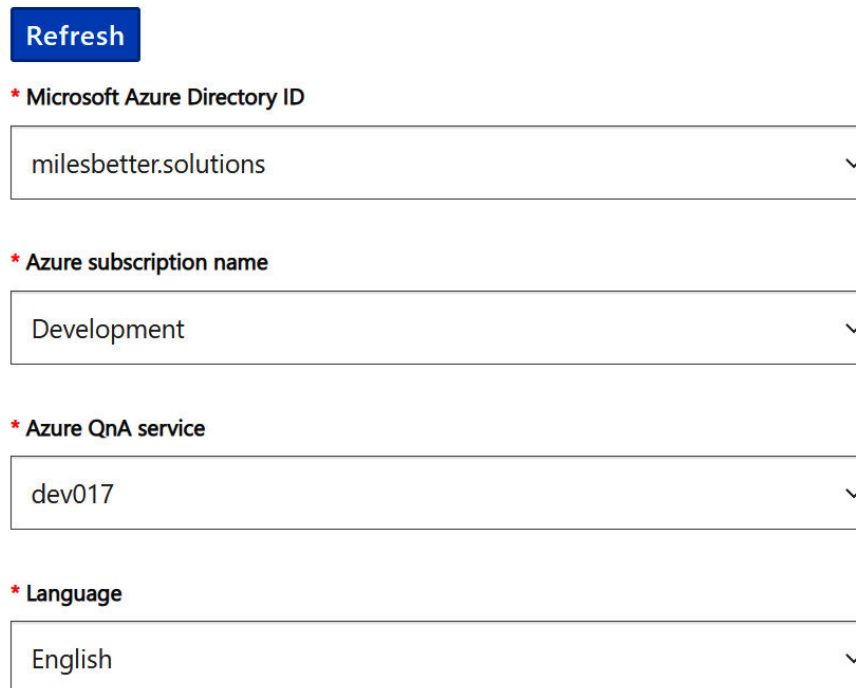
- **Azure Search location:** Choose a location.
 - **Azure Search pricing tier:** Select **Free F**.
 - **Website location:** Chose a location.
 - **App Insights location:** Choose a location.
6. Click **Review + create**.
 7. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
 8. When the deployment is complete, you will receive a notification stating that the deployment succeeded. Return to the **QnA Maker** portal and from **STEP 2**, click **Refresh**.

Task – Configuring QnA Maker

9. From **STEP 2**, select the following as required; **Microsoft Azure Directory ID**, **Azure subscription name**, **Azure QnA service**, and **Language**:

STEP 2 Connect your QnA service to your KB.

After you create an Azure QnA service, refresh this page and then select your Azure service using the options below



The screenshot shows a configuration interface with a blue 'Refresh' button at the top. Below it are four dropdown menus, each with a red asterisk indicating a required field. The first dropdown is labeled '* Microsoft Azure Directory ID' and contains the text 'milesbetter.solutions'. The second is labeled '* Azure subscription name' and contains 'Development'. The third is labeled '* Azure QnA service' and contains 'dev017'. The fourth is labeled '* Language' and contains 'English'. Each dropdown menu has a small downward-pointing chevron icon on the right side.

Figure 5.43 – Connecting the bot service

10. From **STEP 3**, name your KB as you wish.
11. From **STEP 4**, choose a **Chit-chat** setting.

12. From **STEP 5**, click **Create your KB**; an empty QnA will be created:

STEP 5 Create your KB

The tool will look through your documents and create a knowledge base for your service. If you are not using an existing document, the tool will create an empty knowledge base table which you can edit.

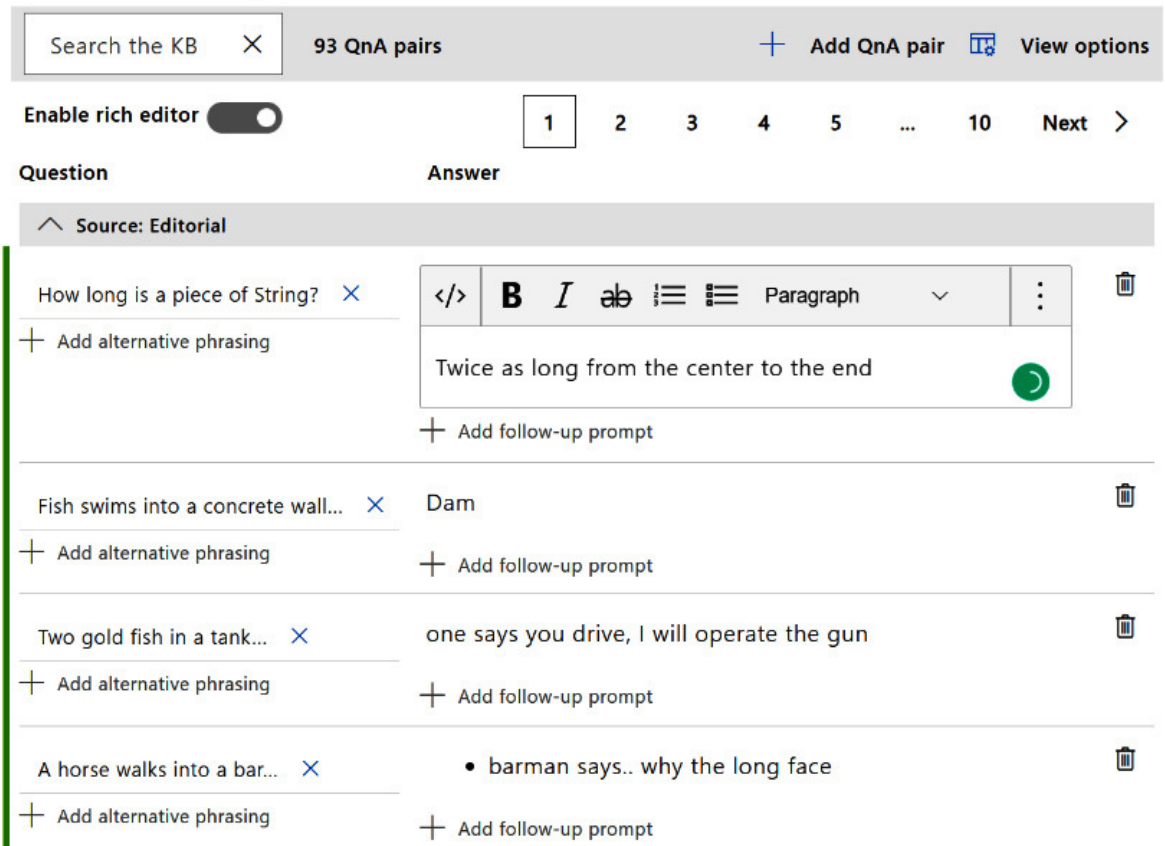


Figure 5.44 – Creating a bot knowledge base

13. From the QnA **Knowledge base** screen, click **Add QnA pair**; this will add a new row to the KB table.

14. In the **Question** cell, enter your question (or input) text; in the **Answer** cell, enter your answer (or response) text:

Knowledge base



The screenshot shows the 'Knowledge base' interface. At the top, there is a search bar labeled 'Search the KB' with a close icon, and a status indicator '93 QnA pairs'. To the right are buttons for '+ Add QnA pair' and 'View options'. Below this is a toggle for 'Enable rich editor' which is turned on. A pagination bar shows '1' selected, followed by '2', '3', '4', '5', '...', '10', and 'Next >'. The main area is a table with two columns: 'Question' and 'Answer'. The first row is expanded, showing a rich text editor for the question 'How long is a piece of String?' and the answer 'Twice as long from the center to the end'. The editor includes a toolbar with icons for bold, italic, underline, list, paragraph, and a refresh button. Below the question and answer cells are links for '+ Add alternative phrasing' and '+ Add follow-up prompt'. The table contains four rows of QnA pairs:

Question	Answer
How long is a piece of String?	Twice as long from the center to the end
Fish swims into a concrete wall...	Dam
Two gold fish in a tank...	one says you drive, I will operate the gun
A horse walks into a bar...	• barman says.. why the long face

Figure 5.45 – Populating the bot knowledge base

15. Repeat *Steps 19* and *20* as required.

16. Click **Save and train** from the top-right toolbar.

Task – Testing the solution

17. From the top toolbar, click -> **Test**.

18. Enter text from the question values you created earlier; the answer value you created will be returned. In our example, the response is the joke punchline; the following are some examples:

NGS Save and train → Test

Published KB ? Start over

how long is a piece of string?

Twice as long from the center to the end Inspect You

Its the way I tell em **(Test)** at 10:32 PM

A horse walks into a bar

- barman says.. why the long face

Inspect You

Its the way I tell em **(Test)** at 10:32 PM

fish swims into concrete wall

Dam Inspect You

Its the way I tell em **(Test)** at 10:33 PM

Two gold fish in a tank

one says you drive, I will operate the gun Inspect You

Its the way I tell em **(Test)** at 10:33 PM

Type your message here ...

Figure 5.46 – Testing the bot knowledge base

19. With that, we have completed this chapter's exercises.

This section covered various hands-on exercises. Now, let's summarize this chapter.

Summary

This chapter provided complete coverage of the AZ-900 Azure Fundamentals exam skills area called *Describe Core Azure Solutions*.

In this chapter, you learned about various skills that will provide you with the confidence to explain and discuss the following aspects with a business or technical audience: *serverless computing, artificial intelligence, Internet of Things, big data and analytics, and DevOps*.

Every organization should be evaluating the solutions referenced in this chapter; this is more important than ever for organizations of any size. There is a need to accelerate innovation, work smarter and not harder, and find ways to reduce costs from shrinking budgets. An organization needs to respond quickly to gain a competitive advantage or get software and systems into the market; release and update cycles of years, months, and weeks are no longer tenable.

For each of the technology areas we have covered in this chapter, the question should no longer be *Why would we want to do this?* but instead, *How do we accelerate adoption quicker?*

We are ushering in a new era of digital leaders; a culture and mindshift are required from the top down to foster teams to work more collaboratively; we need to leave behind the organization's technical debt and shift that on-premises mindset and go beyond DevOps and into a *DigitalOps* culture.

As they say, *you are on the bus or under its wheels*; the DevOps and Data Insights bus has left the depot... *don't get left behind... or under the wheels*.

In this chapter, further knowledge beyond the exam's content was provided to help prepare you for a real-world, day-to-day Azure-focused role.

The chapter concluded with a hands-on exercise section that brought together some of the skills areas that were covered in this chapter.

The next chapter will outline Azure Management tools such as the Azure portal, PowerShell, the Azure CLI, Cloud Shell, and Mobile App. We will also describe the functionality and usage of Azure Advisor, Azure Monitor, and Azure Service Health.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure Fundamentals: <https://docs.microsoft.com/learn/certifications/exams/az-900>
- Exam AZ-900: skills outline: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- Microsoft Learn: Azure Fundamentals – Describe core solutions on Azure: <https://docs.microsoft.com/learn/paths/az-900-describe-core-solutions-management-tools-azure>
- Azure Serverless, Azure Logic Apps, and Azure Functions: <https://docs.microsoft.com/azure/logic-apps/logic-apps-serverless-overview>
- Azure IoT: <https://docs.microsoft.com/azure/iot-fundamentals>
- Artificial intelligence (AI): <https://docs.microsoft.com/azure/architecture/data-guide/big-data/ai-overview>
- Cognitive Services: <https://docs.microsoft.com/learn/paths/explore-computer-vision-microsoft-azure/>
- Big data and analytics: <https://azure.microsoft.com/solutions/big-data/#solution-architectures>
- Azure DevOps: <https://docs.microsoft.com/azure/devops/?view=azure-devops>

Skills check

Challenge yourself with what you have learned in this chapter by answering the following questions:

1. Explain how serverless is different from IaaS and PaaS (the key differentiators).
2. Explain the benefits of serverless.
3. Explain what Azure functions are.
4. Explain what Azure logic apps are and how they are different from Azure functions.
5. Explain what artificial intelligence is and what value it provides an organization with.
6. Explain how AI, ML, and DL are connected.
7. Explain the differences between Azure Machine Learning and Azure Cognitive Services.
8. List and explain the three core elements of an IoT solution.
9. List and explain at least three use case scenarios for IoT solutions.
10. Explain the difference between Azure IoT Central and IoT Hub.
11. Explain Azure Sphere.
12. Explain the term big data.
13. Explain the Modern Data Warehouse approach and ELT versus ETL.
14. Explain what is meant by DevOps and what its value is to an organization.
15. Explain what GitHub is.
16. Explain the difference between Git and GitHub.
17. Explain what GitHub Actions is.
18. Explain the overlap between GitHub and Azure DevOps.
19. Explain what Azure DevTest Labs is.
20. Explain how Azure DevTest Labs fits in with a DevOps practice.

Chapter 6: Azure Management Tools

In [Chapter 5](#), *Core Azure Solutions*, you learned about **serverless computing**, **artificial intelligence**, the **Internet of Things**, **big data** and **analytics**, and **DevOps**.

This chapter will outline the *management tools* available in Azure, including the **Azure portal**, **Azure PowerShell**, the **Azure CLI**, **Cloud Shell**, the **Azure mobile app**, **Azure Advisor**, **Azure Monitor**, and **Azure Service Health**.

This chapter aims to provide coverage of the AZ-900 Azure Fundamentals *Skills Measured* section called *Describe management tools on Azure*.

By the end of this chapter, you will have learned about various skills to be able to describe the functionality and usage of the following:

- The Azure portal, Azure PowerShell, the Azure CLI, Cloud Shell, and the Azure mobile app
- Azure Advisor
- Azure Monitor
- Azure Service Health

To support your learning with some practical skills, we will also look at a hands-on example and usage of some of the tools and resources covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – installing Azure PowerShell
- Exercise 2 – installing the Azure CLI
- Exercise 3 – creating resources using PowerShell from Cloud Shell
- Exercise 4 – creating resources using the Azure CLI from Cloud Shell
- Exercise 5 – exploring Azure Service Health

In addition, this chapter's goal is to take your knowledge beyond the exam objectives so that you are prepared for a real-world, day-to-day, Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will need the following:

- An Azure subscription that has access to create and delete resources in the subscription. If you do not have an Azure subscription, you can create a free Azure account at <https://azure.microsoft.com/free>.
- Access to an internet browser; you will log in to the Azure portal: <https://portal.azure.com>.
- Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.
- A Windows 10 device with *PowerShell 7.x* or *later* and rights to install *Azure PowerShell* and the *Azure CLI*: <https://docs.microsoft.com/powershell/scripting/install/installing-powershell-core-on-windows>.

Azure portal

The **Azure portal** is a browser-based **graphical user interface (GUI)** console for interacting with Azure resources.

The portal is designed for self-service and is the most common method for creating and managing your Azure environments. It is the quickest way for anybody new to Azure to get started and carry out simple tasks; a *desktop app* is also available to provide the same user interface experience.

There are other ways to interact with Azure services and resources; if you want to create and manage more complex resources or perform automation tasks, you can use a **command-line interface (CLI)** such as *PowerShell* or **Bourne Again Shell (Bash)**. Finally, you can use the *Azure mobile app* if you need quick visibility and must interact with your Azure resources from anywhere, anytime. In this chapter, you will learn about how to use each of these to interact with your Azure resources. You will also gain some practical skills with some of these management tools in the *Exercise* section. The Azure portal is public-facing and can be accessed at <https://portal.azure.com>.

The Azure portal provides users with a quick and straightforward way to access Azure resources, often through dashboard views or high-level overview visualizations; these can also be exported in various formats, such as reports and Excel, based on the resource type.

The Azure portal allows users to *create*, *change*, and *delete* resources directly within the portal as a seamless single user interface experience. The Azure portal can also be accessed from a *mobile device*; you can use the *Azure mobile app* (*iOS* and *Android*) to do so, which we will cover in the *Azure mobile app* section later in this chapter.

Being a web-based console, the portal can be accessed from anywhere if you have an internet connection and from all modern desktops, laptops, and tablet devices (*running Windows, Android, and so on*). The only limitation is that your browser must have *JavaScript* enabled, and your operating system must be capable of running one of the following supported browsers:

- Microsoft Edge (latest version)
- Safari (latest version, Mac only)

- Chrome (latest version)
- Firefox (latest version)

The portal user experience can be customized through personal settings. These settings can be accessed by clicking the *gears icon* on the top right of the portal toolbar; the following settings are available:

- *Directories + subscriptions*
- *Appearance + Startup views*
- *Language + region*
- *My information*
- *Signing out + notifications*

In addition to the preceding settings, customized dashboards can be created.

All tasks that are carried out in the portal use the underlying **Azure Resource Manager (ARM)** API. The ARM API performs all the activities in the background and will notify you of progress and whether any tasks have been completed or failed.

Full deployment and activity logging is provided, which provides all the activities/tasks that a user and the Azure platform initiate; this can be helpful in troubleshooting issues and for audit and compliance purposes.

Governance can also be put in place through Azure Policy, and access controls can be put in place through **role-based access control (RBAC)**, both of which you will learn about in more detail in [Chapter 9, Azure Governance](#); access controls can prevent users from *adding, changing, or deleting* resources and locks can also be added to resources to prevent accidental deletion.

This section looked at the Azure portal and covered some of its functionalities. The following section will look at Azure PowerShell.

Azure PowerShell

PowerShell is a cross-platform *CLI* management tool for interacting with Azure resources. This tool can be used instead of the *GUI* of the Azure portal or desktop/mobile apps for creating and managing your Azure environments. Azure PowerShell is a module that's installed as part of Windows PowerShell; the *Az PowerShell* module contains a set of *cmdlets* (*pronounced as commandlets*) for *PowerShell* that allow you to manage resources from within PowerShell directly at the *CLI* without the need to access the portal.

PowerShell is a popular management tool when the focus is on Windows systems and is designed for complex automation tasks. It can be used interactively, meaning commands can be entered manually by typing them directly into the shell command prompt. Providing cross-platform support (from *PowerShell Core 6.x* and *PowerShell 7.x*) means that you can install and use the PowerShell *Az module* on Windows, Linux, or macOS to interact with your Azure environments.

You can also access the CLI through the portal using Cloud Shell, which provides access to the Az PowerShell module cmdlets; we will cover this in the *Azure Cloud Shell* section.

PowerShell 7 is the recommended version of PowerShell to use with Az PowerShell on all platforms; that is, Windows, macOS, and Linux. The Az module is based on *the .NET Standard library*.

PowerShell 7.0.6 or *7.1.3* or *later* is required for *Az module 6.0.0* and *later*; however, *PowerShell 5.1* is still supported.

To use the *PowerShell Az module*, you must install it (*it is pre-installed in Azure Cloud Shell*). The preferred installation method for the Az module is to use the `Install-Module` cmdlet, and the recommended *installation scope* is for the *current user* only. You will learn how to install the Azure PowerShell module and some simple PowerShell commands in the *Hands-on exercise* section of this chapter.

The syntax (*the structure or placement of words, or commands in this case*) of PowerShell commands uses the verb-noun pair format, and the data (*response or answer to your question*) that's *returned* is called an object.

The *verb* (*first part*) of the pair refers to the action that the cmdlet must perform, such as **Get**, **Set**, **Show**, **Find**, **New**, and **Resize**; the *noun* (*second part*) of the pair refers to the entity that the action is performed is; that is, *do this to that*.

A simple example of this is `Get-<Verb>`, which can be used to get the following approved PowerShell *command verbs*:

- `Connect-AzAccount` can be used to log in to Azure.
- `Get-AzResourceGroup` will list all resource groups.
- `NewAzResourceGroup` will create a new resource group.
- `New-AzVM` will create a new VM.

You can also check the PowerShell version by running the following command from within PowerShell:

```
$PSVersionTable.PSVersion
```

These are just a few PowerShell examples. We will look at some of these commands in the hands-on exercise in this chapter.

PowerShell can also run *PowerShell scripts*; these scripts contain PowerShell cmdlets and code and can only be run (*executed*) from within PowerShell. Scripts automate repetitive or complex tasks that have many steps and actions to be performed against many different entities. In this case, a series of commands can be assembled in the syntax format of the shell being used, and the script can then be executed by issuing a single command at the PowerShell prompt.

This section looked at the PowerShell CLI as an Azure management tool. The following section looks at the Azure CLI.

The Azure CLI

The **Azure CLI** is a cross-platform *CLI* management tool for interacting with Azure resources; this can be used instead of the *GUI* for creating and managing your Azure environments.

Providing cross-platform support means you can install and use it on Windows, Linux, or macOS. It draws parallels to Bash shell scripting and is a popular management tool choice when the focus is on Linux systems and is designed for complex and automation tasks; it is written in *Python*.

Much like the PowerShell Az module, commands can be executed using interactive commands; they are directly from the shell prompt or using scripts to automate repetitive or complex tasks that have many steps and require actions to be performed against many different entities.

In this case, a series of commands can be assembled in the syntax format of the shell being used, and the script is then executed by issuing a single command at the shell prompt. This is done within the shell of the OS you have installed the Azure CLI on, such as `cmd.exe` for Windows or *Bash* for Linux and Mac.

IMPORTANT NOTE

Cloud Shell is a Microsoft-provided Shell environment hosted for you on Ubuntu Linux containers (that Microsoft manages and maintains, that you do not pay for); you can think of this as Shell Environment as a Service. The quickest way to start using the Azure CLI is by running it in an Azure Cloud Shell; this can be used instead of a local Shell environment hosted on Windows, Linux, or macOS machines. Using the Azure CLI in Cloud Shell will be covered in the Azure Cloud Shell section of this chapter.

The Azure CLI can be installed for each OS like so:

- Windows: *MSI installer package file format*
- Ubuntu, Debian Linux: *apt package manager*
- RHEL, Fedora, CentOS: *dnf package manager*
- openSUSE, SLES: *zypper package manager*
- macOS: *homebrew package manager*

You can start using the Azure CLI to perform creation and management tasks as you would do in the portal or via the PowerShell Az module (*Azure PowerShell*).

To start using the Azure CLI, first, you need to sign in. You can use the following command, which will load an Azure sign-in page; this is referred to as an *interactive* sign-in:

```
az login
```

The syntax (*the structure or placement of words or commands, in this case*) of the Azure CLI follows a similar but different pattern to PowerShell; it uses the following format pattern:

```
az <command group> <parameters>
```

The following are the same example tasks we looked at in the *PowerShell* section, but this time using the *Azure CLI* syntax:

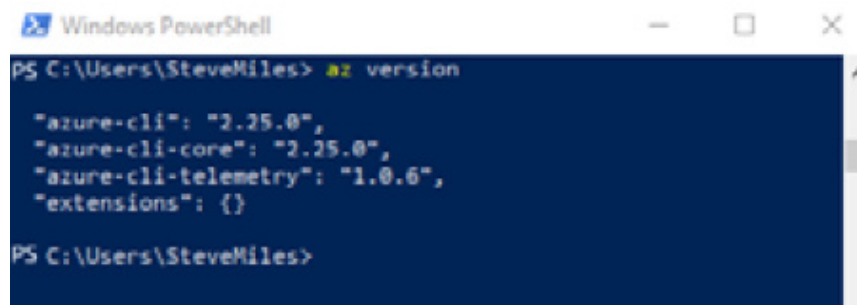
- **az login** can be used to log in to Azure.
- **az group list** will list all resource groups.
- **az group create** will create a new resource group.
- **az vm create** will create a new VM.

These are just a few Azure CLI examples. We will look at some of these commands in the hands-on exercise in this chapter.

After installing the Azure CLI, you can check its version by running the following command from either the Windows Command Prompt or PowerShell:

```
az version
```

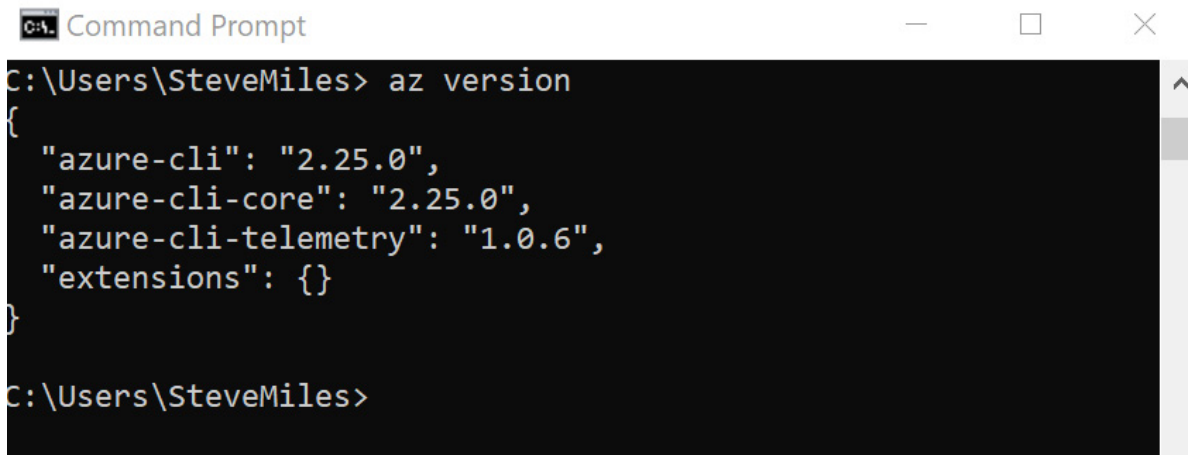
The following screenshot shows the command output for *Windows PowerShell*:



```
Windows PowerShell
PS C:\Users\SteveMiles> az version
{"azure-cli": "2.25.0",
 "azure-cli-core": "2.25.0",
 "azure-cli-telemetry": "1.0.6",
 "extensions": {}}
```

Figure 6.1 – The azure-cli version command from PowerShell

The following screenshot shows the command output for the *Windows Command Prompt*:

A screenshot of a Windows Command Prompt window. The title bar reads 'C:\ Command Prompt'. The prompt shows the command 'az version' being executed. The output is a JSON object: { "azure-cli": "2.25.0", "azure-cli-core": "2.25.0", "azure-cli-telemetry": "1.0.6", "extensions": {} }. The prompt then returns to 'C:\Users\SteveMiles>'.

```
C:\Users\SteveMiles> az version
{
  "azure-cli": "2.25.0",
  "azure-cli-core": "2.25.0",
  "azure-cli-telemetry": "1.0.6",
  "extensions": {}
}
C:\Users\SteveMiles>
```

Figure 6.2 – The azure-cli version command from the Windows Command Prompt

This section looked at the Azure CLI as an Azure management tool. The following section will look at Azure Cloud Shell as a browser-based alternative to a Shell environment from a physical or virtual machine.

Azure Cloud Shell

Azure Cloud Shell is a cross-platform, interactive, hosted Shell and scripting environment that Microsoft provides hosted on *Ubuntu containers*; you can think of this as a *Shell environment as a Service*.

Cloud Shell enables a browser-based *CLI*. The benefit of using Cloud Shell means that you do not need to download, install, or update any CLI management tools in a local Shell environment on a device or machine. Being a cross-platform management tool, all you need is a browser to run shell commands and the *PowerShell Az* module.

Imagine that you were to move between devices; you might not have access to the necessary CLI tools and you may not have the PowerShell modules or updates. This means that your scripts may fail to run, or your interactive commands error as they are intended for a different version than you have installed locally. However, with Cloud Shell, wherever you have access to a browser, you have access to a CLI; you will always have a consistent and up-to-date experience. You can also use Cloud Shell from the Azure mobile app so that you have a Shell environment from anywhere, at any time.

Cloud Shell provides two Shell environments so that you can choose the one that best suits your requirements, as follows:

- Bash; *with the Azure CLI installed*
- PowerShell; *with the Azure PowerShell module installed*

Cloud Shell can be accessed directly within the Azure portal via the *code icon* at the top of the portal toolbar, as shown in the following screenshot:

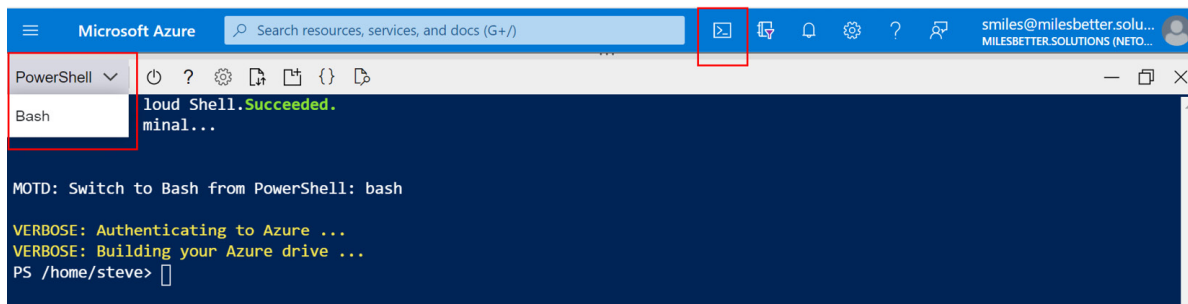


Figure 6.3 – Azure Cloud Shell with the PowerShell interface via the Azure portal

You can select your required Shell environment from a dropdown, depending on whether you wish to run *Bash* or *PowerShell* commands. You can drag the Cloud Shell pane up and down to resize it and have a split view, as shown in the following screenshot:

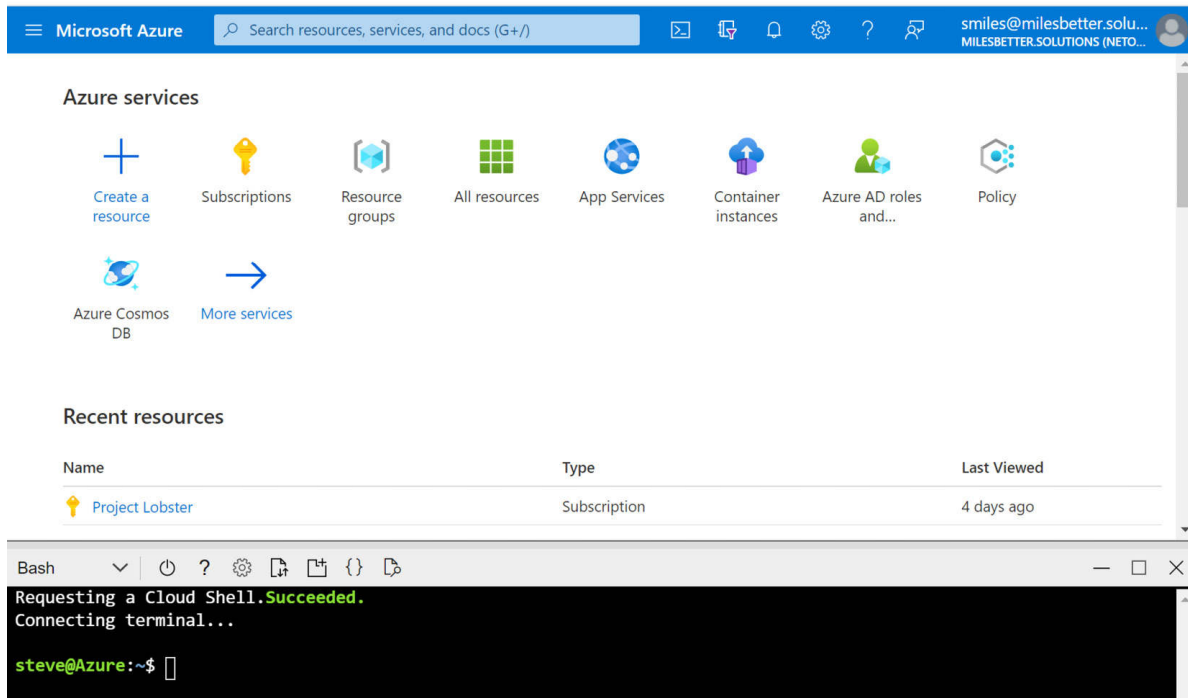


Figure 6.4 – Azure Cloud Shell with the Bash interface via the Azure portal

Cloud Shell is also available via a standalone browser-based shell, without the need to access it via the Azure portal; this standalone browser-based shell experience can be accessed from the following URL: <https://shell.azure.com>.

As we mentioned in the *Azure CLI* section of this chapter, the Azure CLI can be accessed via either the *PowerShell* or *Bash Shell* environment within Cloud Shell; this is pre-installed, unlike running it on your device, and is always available wherever you have access to a browser; this makes it portable by nature, not like an install that has been carried out on every device that you may access.

This section looked at Cloud Shell as an Azure management tool. The following section will look at Terraform on Azure.

Terraform on Azure

Terraform is available natively in Azure Cloud Shell and is HashiCorp's *declarative*-based open source **Infrastructure as Code (IAC)** tool.

It allows the life cycle of *codified infrastructure* to be managed by declaring the infrastructure components to be provisioned with cloud providers in descriptive configuration files; this allows us to automate the ability to provision, change, and tear down any infrastructure components.

This *codified infrastructure* defined in `.tf` file type text files can be controlled and governed in a manner that is repeatable and predictable, and safe for any existing environments that this defined infrastructure is to be deployed into; this can be achieved by previewing and validating any impact this will have on any infrastructure changes before they are committed as adds, changes, or removes. Since they're text-based configuration files, it is easy to integrate them with source control (version control) systems such as Git and Azure DevOps.

Configuration files are created using the Terraform language, which uses the **HashiCorp Configuration Language (HCL) syntax**, though JSON syntax (*easier for machines to parse*) can also be used in the files. The benefit of the `.tf` format is that it is both machine- and human-readable.

As shown in the following screenshot, you can check the version of Terraform that's been installed in Azure Cloud Shell by running the `terraform -version` command:

```
Bash  v | ? ? ? ? ? ? ? ?  
Requesting a Cloud Shell.Succeeded.  
Connecting terminal...  
  
Welcome to Azure Cloud Shell  
  
Type "az" to use Azure CLI  
Type "help" to learn about Cloud Shell  
  
steve@Azure:~$ terraform -version  
Terraform v1.0.3  
on linux_amd64
```

Figure 6.5 – Terraform version

As shown in the following screenshot, you can check the available commands by running the `terraform -h` command:

```
Bash  v | ? ? ? ? ? ? ? ?  
steve@Azure:~$ terraform -h  
Usage: terraform [global options] <subcommand> [args]  
  
The available commands for execution are listed below.  
The primary workflow commands are given first, followed by  
less common or more advanced commands.  
  
Main commands:  
  init           Prepare your working directory for other commands  
  validate       Check whether the configuration is valid  
  plan           Show changes required by the current configuration  
  apply          Create or update infrastructure  
  destroy        Destroy previously-created infrastructure  
  
All other commands:  
  console        Try Terraform expressions at an interactive command prompt  
  fmt            Reformat your configuration in the standard style  
  force-unlock  Release a stuck lock on the current workspace
```

Figure 6.6 – Terraform commands

This section looked at Terraform on Azure. In the next section, we will at the Azure mobile app, which you can use to interact with your Azure resources from wherever you have access to a mobile device or tablet.

Azure mobile app

The **Azure mobile app** allows you to monitor and interact with your Azure resources via an *Android* or *iOS* device and stay connected anytime, from anywhere.

The app can be downloaded for free from the Apple App Store and Google Play; it is optimized for smartphones and works on tablets. The Azure mobile app supports *iOS 11.0 and later* and *Android 6.0 and later*.

The Azure mobile app is great when you cannot access a computer; maybe you only have a mobile or a tablet. Still, you need visibility to resources to be able to check their status, health, and the ability to diagnose and fix issues quickly, as well as being able to access Azure Cloud Shell. You will always have a portable way to access a Shell environment and a CLI. The following screenshot shows some of the Azure mobile app screens:

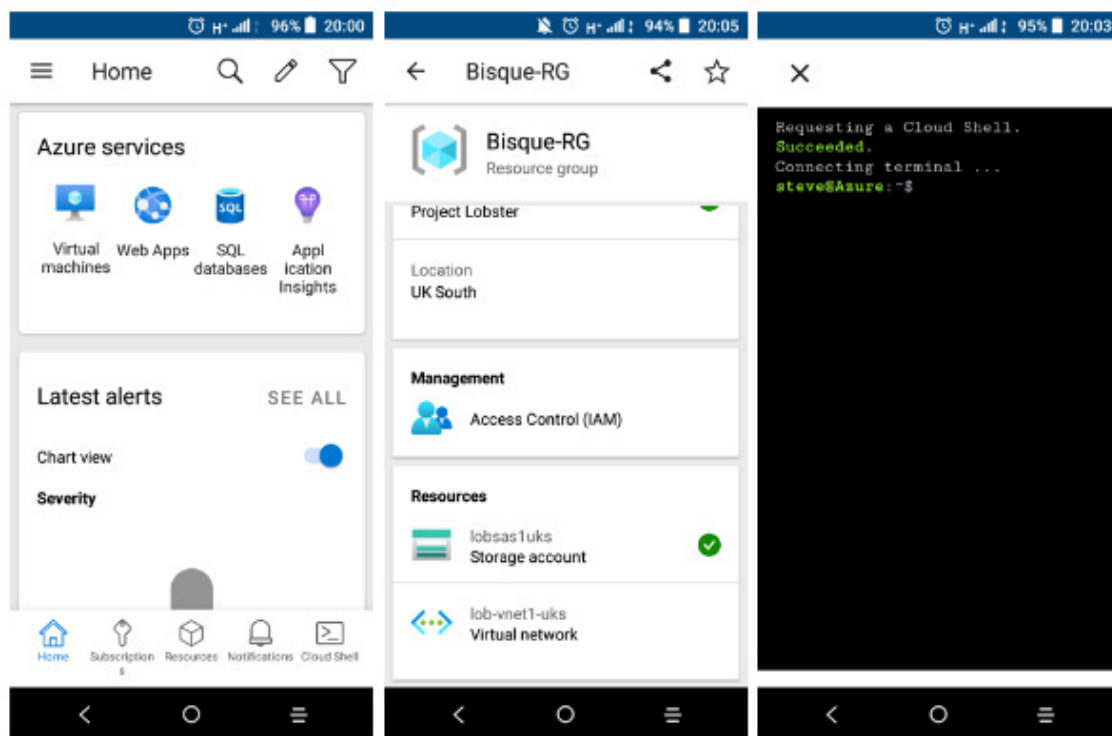


Figure 6.7 – Azure mobile app interface

The preceding screenshot shows the Mobile App experience. More information on the Azure mobile app can be found at <https://azure.microsoft.com/get-started/azure->

[portal/mobile-app](#).

This section looked at using the Azure mobile app to interact with Azure resources via an Android or iOS device. The following section looks at Azure Advisor.

Azure Advisor

Azure Advisor is an included, no-cost service that provides advice on optimizing your Azure resources. It provides personalized and actionable *best practice* recommendations based on usage analysis and can be accessed directly within the portal.

There is a great deal of emphasis placed on using Azure Advisor and taking actions based on the recommendations. As such, Azure Advisor will be displayed when you log in to the Azure portal, as shown in the following screenshot:

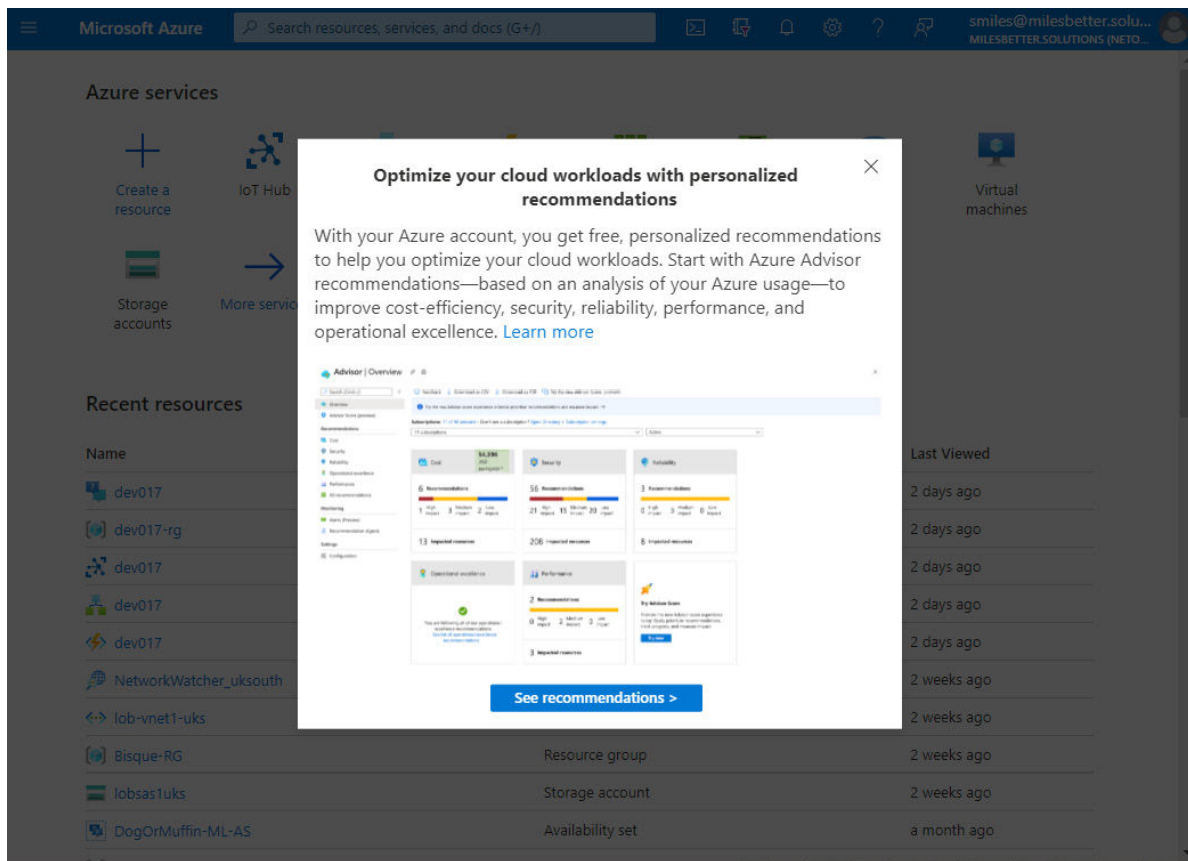


Figure 6.8 – Azure Advisor prompt

Azure Advisor recommendations are available across the following categories; the URLs point to the complete list of recommendations available for each category:

- **Cost:** Recommendations provided for optimizing and reducing costs: <https://docs.microsoft.com/azure/advisor/advisor-cost-recommendations>.

- **Security:** Recommendations provided to protect against vulnerabilities and threats: <https://docs.microsoft.com/azure/advisor/advisor-security-recommendations>.
- **Reliability** (formerly **High Availability**): Recommendations provided for the availability, resilience, and continuity of your workloads: <https://docs.microsoft.com/azure/advisor/advisor-high-availability-recommendations>.
- **Operational Excellence:** Recommendations provided for implementing best practices, resources manageability, service health, and governance: <https://docs.microsoft.com/azure/advisor/advisor-operational-excellence-recommendations>.
- **Performance:** Recommendations provided for optimizing a workload's responsiveness to demand: <https://docs.microsoft.com/azure/advisor/advisor-performance-recommendations>.

The following screenshot shows the **Azure Advisor | Overview blade**, which displays these recommendations in a dashboard view:

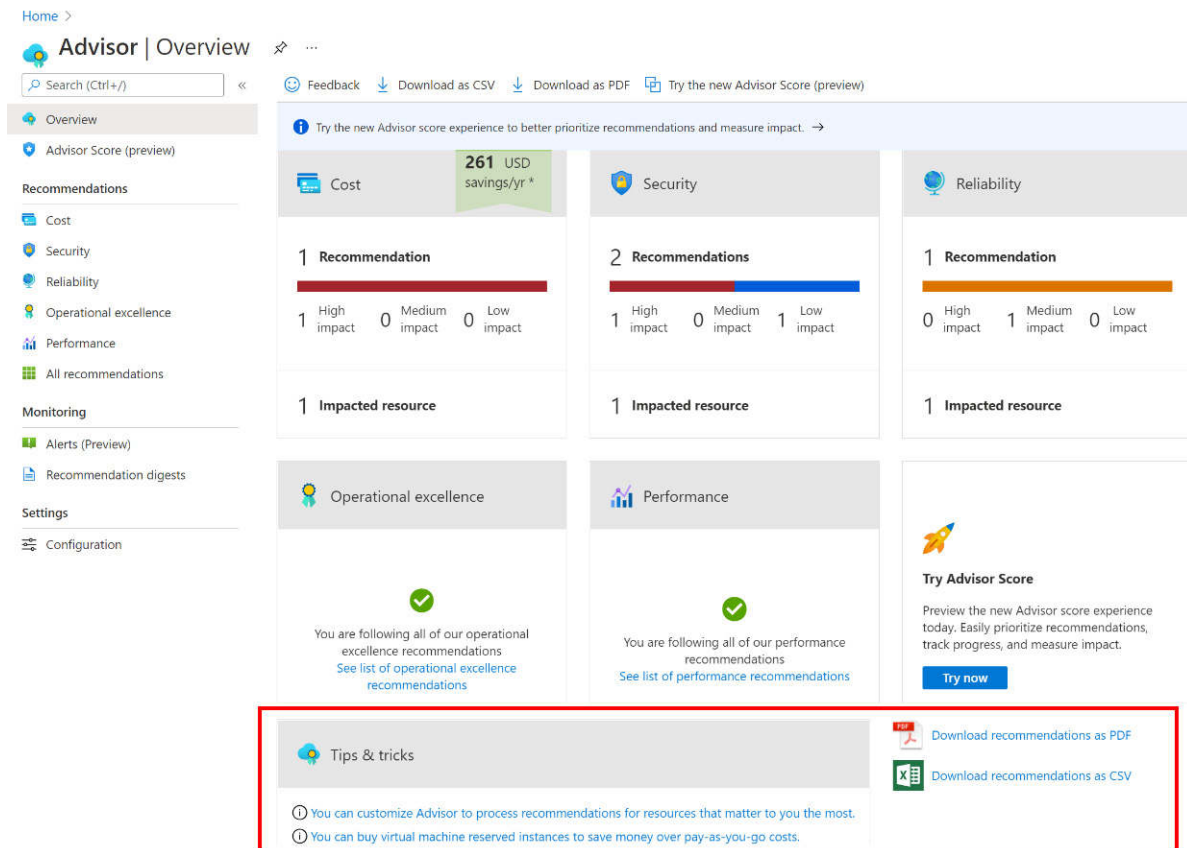


Figure 6.9 – Azure Advisor recommendations

There is a *Tips & tricks* section at the bottom of the overview blade that helps you get the most out of Azure Advisor. You can also download the recommendations in PDF and Excel format.

Azure Advisor recommendations are based on the approach of notifying users of aspects that are not implemented rather than things that are implemented; for example, Azure Advisor will generate a list of the following:

- VMs that *are not* protected by a **Network Security Group (NSG)**
- VMs that *do not* have backup enabled, updates applied, a premium SSD when capable, and so on
- Storage accounts that *do not* have soft delete enabled, and so on

You can click on each recommendation category to get more information on the recommendations and the actions that can be taken.

Once any *security recommendations* have been implemented, you will see that the organization's *Secure Score* will have *increased*. However, there is no requirement that any recommendations *must* be implemented to be supported by Microsoft, only that they *should* be implemented. There is also no timescale as to *when and if* you decide to implement these recommendations; they can be dismissed or *postponed*.

This section looked at Azure Advisor as a recommendation engine to aid in optimizing your Azure environments. The following section looks at Azure Monitor, which can help you gain visibility and insights into all the resources that are created in your Azure environments.

Azure Monitor

You can't control what you can't see.

Azure Monitor is an included service that provides actionable insights into the health, availability, and performance of Azure and on-premises environments by collecting and analyzing logs and metrics (*telemetry*). It allows you to find and fix problems faster, optimize your workload's performance, and then provide actions to remediate and alert; it provides all these insights from within the portal.

Azure Monitor collects resources and platform data from the following data sources:

- PaaS resources such as applications and databases
- IaaS resources, such as VMs, containers, virtual desktops, databases, storage, and many more
- On-premises resources

With Azure Monitor, you can create alerts that notify somebody, such as an administrator or resource owner, when certain conditions are triggered. This could be when a resource exceeds a set threshold, when a resource is stopped, such as a VM, or when a resource is deleted, such as a storage account. The following diagram aims to visualize the Azure Monitor service and its three core elements; that is, *data sources*, *data stores*, and *data functions*:

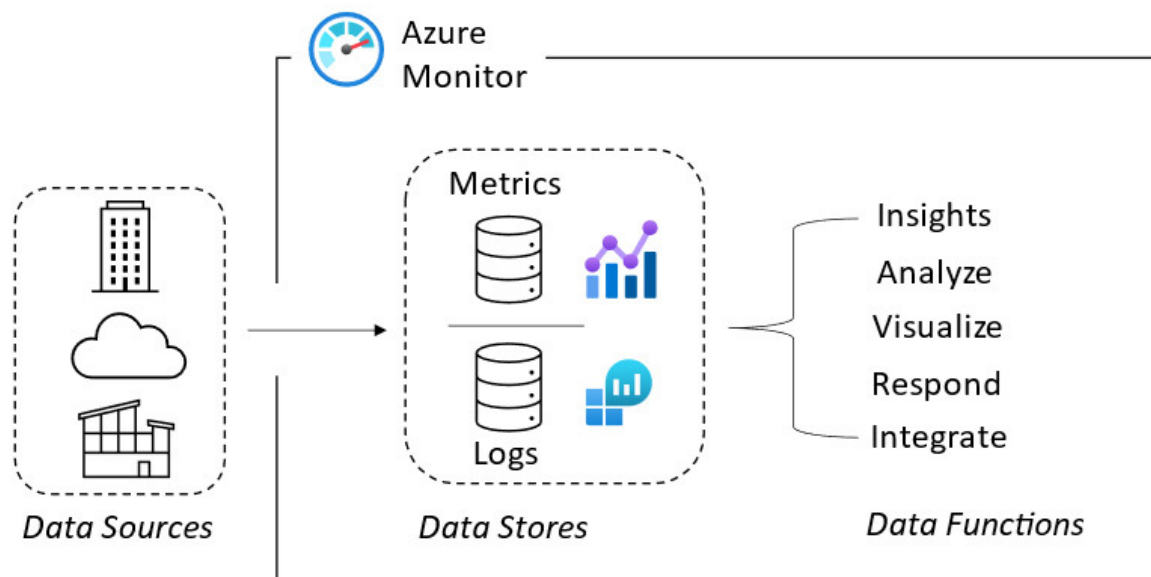


Figure 6.10 – Azure Monitor components architecture

Two fundamental types of data are used by the Azure Monitor service; these are *log* and *metric* data. The monitoring data sources populate these two data stores; the data functions then consume these data stores. The data stores support near-real-time eventing.

The differences between *logs* and *metrics* are as follows:

- Logs:** These record the *activities* of a data source representing some form of action that was taken against the resources. This could include capturing a login, a create/delete/update action, list storage account keys, whether a policy was run, and whether an automation job started; this data is presented in column format and utilizes **Log Analytics**:



Figure 6.11 – Azure Monitor log data

- Metrics:** These record the performance and consumption of a data source and represent the meters and counters being triggered. Metrics can capture the utilization of CPU, memory, network bandwidth, or disk throughput and it is presented in chart format and uses **Services Metrics**:



Figure 6.12 – Azure Monitor metric data

Azure Monitor, which we looked at in this section, is *better together* with **Azure Health Service**, which we will look at in the following section.

Azure Service Health

Azure Service Health is an included, no-cost service that provides a personalized view of the health of all your Azure resources; it provides guidance and notifications, such as planned maintenance and other advisories, on resource health that are specific to you.

These actionable insights are provided directly within the portal as a subset of the *Azure Monitor* service; this allows you to be alerted on notifications or a health status change so that you can evaluate the situation and take any actions you deem necessary. In addition, you can download reports and **root cause analyses (RCAs)**.

The difference between *Azure Service Health* and the *Azure status page* is that the status page is a public-facing website that requires no login. It has a global view of all the services across all regions; it is useful to get a quicker and bigger picture of incidents that have a widespread impact. The Azure status page can be accessed at <https://status.azure.com/status>.

The following screenshot shows the **Azure status** page, which provides a link to **Azure Service Health** so that you can view issues specific to your resources that may be impacted. Upon doing this, you will be redirected to sign in to the Azure portal:

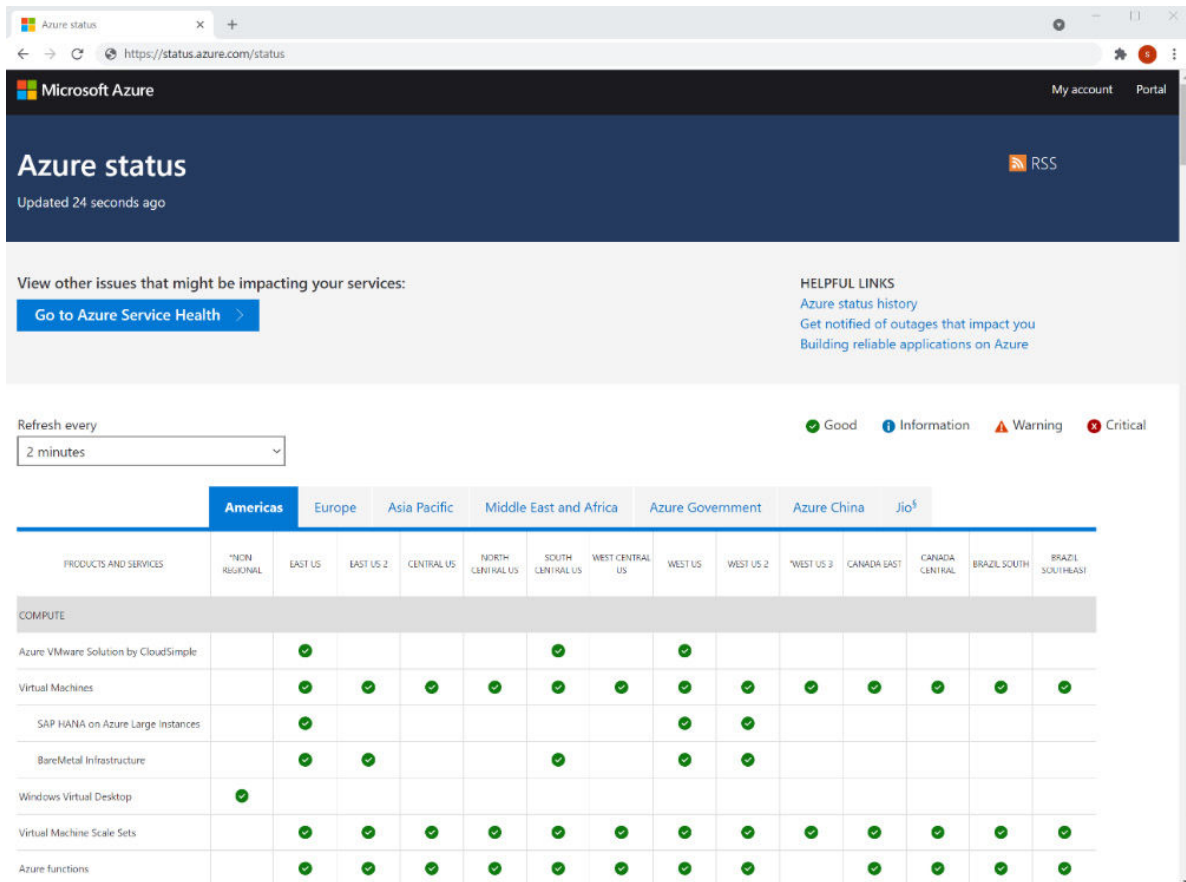


Figure 6.13 – The Azure status page

In this section, we looked at Azure Service Health, which concludes us looking at the individual Azure management tools that can be utilized. In the next section, we will look at the Azure GUI and CLI tools by completing a thought exercise that looks at different scenarios and considerations.

Thought exercise

In this exercise, we will look at our fictitious company, *MilesBetter Pizza*.

Due to the success of the online pizza delivery service and app, they now need to look at the management tools and understand when to choose one tool over another.

As we have learned, there are two approaches to management tools: *GUI* and *CLI*. Which one you choose is *horses for courses*; that is, better tasks are better suited to different tools.

Whether you use the browser-based portal, the desktop, or the Azure mobile app, the GUI approach is the most common way to interact with Azure resources. It's simple, intuitive, and has the quickest time to value and the least skill level entry to be productive in creating and managing resources.

The *GUI approach* does not provide any way to automate repetitive tasks. For example, to set up 15 or 150 VMs, you need to create them one by one by clicking through and completing the wizard each time; this can be time-consuming and error-prone, and allows for configuration drift.

The *CLI approach* allows repetitive tasks and scripting to be automated. Azure offers the flexibility to choose the CLI tool and the Shell environment that suits the user best for the task they need to do at that moment. Users have the choice of using a local shell on the device terminal they wish to execute CLI commands on or use the browser-based shell environment that Azure provides with Cloud Shell. Irrespective of the terminal environment they wish to use to run the commands, they can choose which CLI tool to use; they may choose to use a Bash experience through the Azure CLI for Linux users and PowerShell for Windows users.

Given that there is some parity regarding how these tools function, other factors must be considered when choosing a given task or scenario:

- **Automation:** Do you need to repeat a set of repetitive or complex tasks? PowerShell and the Azure CLI support this, while the GUI portal does not.
- **Complexity and time:** Automation shouldn't be more complex than the task it is replacing; sometimes, a quick, non-complex manual portal or mobile app task/action is appropriate.

- **Learning curve and time:** Do you need to complete a task quickly without learning new commands or syntax? PowerShell and the CLI require you to know the detailed syntax, and if you're not using Cloud Shell, your local terminal device will require you to install the necessary components before you can start. Sometimes, a quick time to value, non-complex manual portal, or mobile app task/action is appropriate here.
- **Team skillset:** Does the team have existing expertise? Do they come from a developer background where they write code and are more comfortable with a command-line and scripting interface (*imperative tools*), or do they come from an admin, engineering, or business analyst background and prefer a more visual, low-code/no-code (*declarative tools*) approach?

While outside of the exam objectives, you should consider that while Azure natively gives you access to tools that allow you to create, configure, and manage Azure resources, there are also third-party tools that should also be considered, and you should evaluate where their use may be appropriate. Some examples of some of the most common third-party tools that you will not be required to know about for the exam objectives, but will almost certainly come across, are *Chef*, *Puppet*, and *Ansible*; we looked at *Terraform on Azure* earlier in this chapter, which can be used as an integrated Azure tool.

The following diagram visualizes the Azure management tools approach:

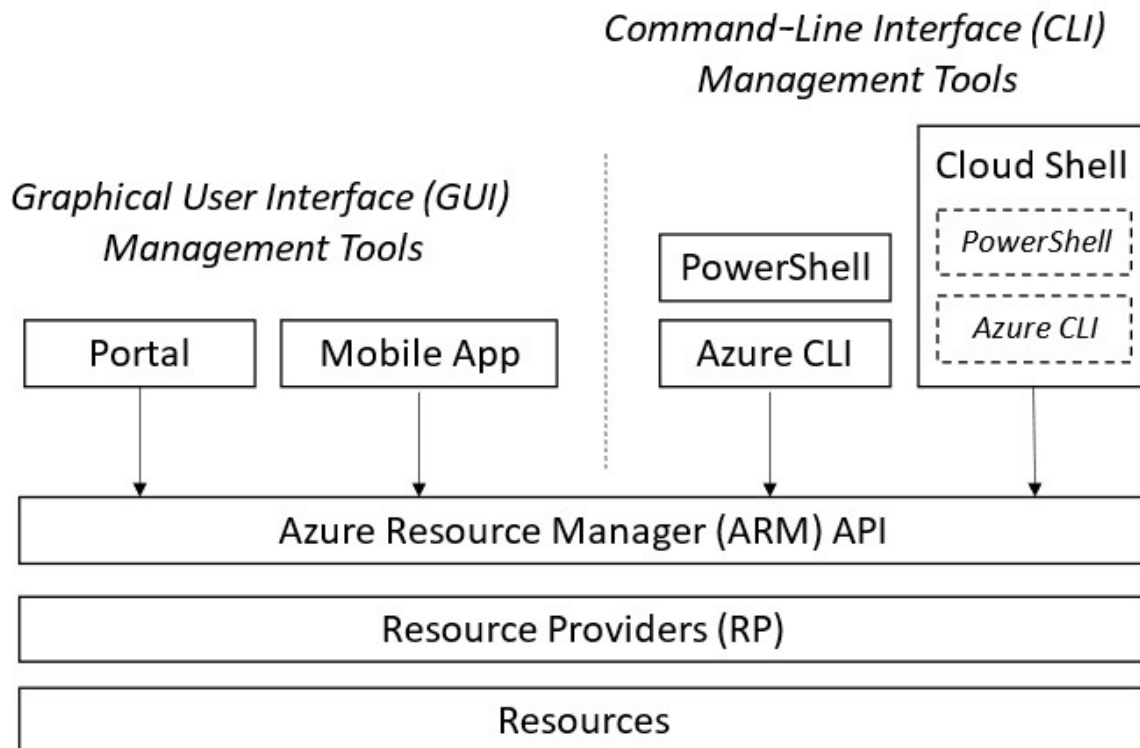


Figure 6.14 – Azure management tools

In this section, we looked at the Azure GUI and CLI tools by conducting a thought exercise and looked at the different scenarios and considerations of when you may choose one management tool over another. In the next section, we will look at some hands-on exercises to help you build on the skills you learned about in this chapter.

Hands-on exercise

To support your learning with some practical skills, we will look at some of the tools that were covered in this chapter by completing some hands-on exercises.

The following exercises will be carried out:

- Exercise 1 – installing Azure PowerShell
- Exercise 2 – installing the Azure CLI
- Exercise 3 – creating resources using PowerShell from Cloud Shell
- Exercise 4 – creating resources using the Azure CLI from Cloud Shell
- Exercise 5 – exploring Azure Service Health

Getting started

To get started with these hands-on exercises, you can use an existing account that you have created as part of the exercises for any chapter in this book. Alternatively, you can create a free Azure account by going to <https://azure.microsoft.com/free>.

This free Azure account provides the following:

- 12 months of free services
- \$200 credit to explore Azure for 30 days
- 25+ services that are always free

Exercise 1 – installing Azure PowerShell

In this section, you'll learn how to install the *Azure PowerShell* module.

The following steps must be carried out on the OS of a machine you have admin access to; this could be physical or virtual. We are using a Windows 10 device for this exercise – *Windows 10 Pro 20H2* using *PowerShell 7* for reference:

1. On your device, search for **Windows PowerShell 7** and click **Open**.

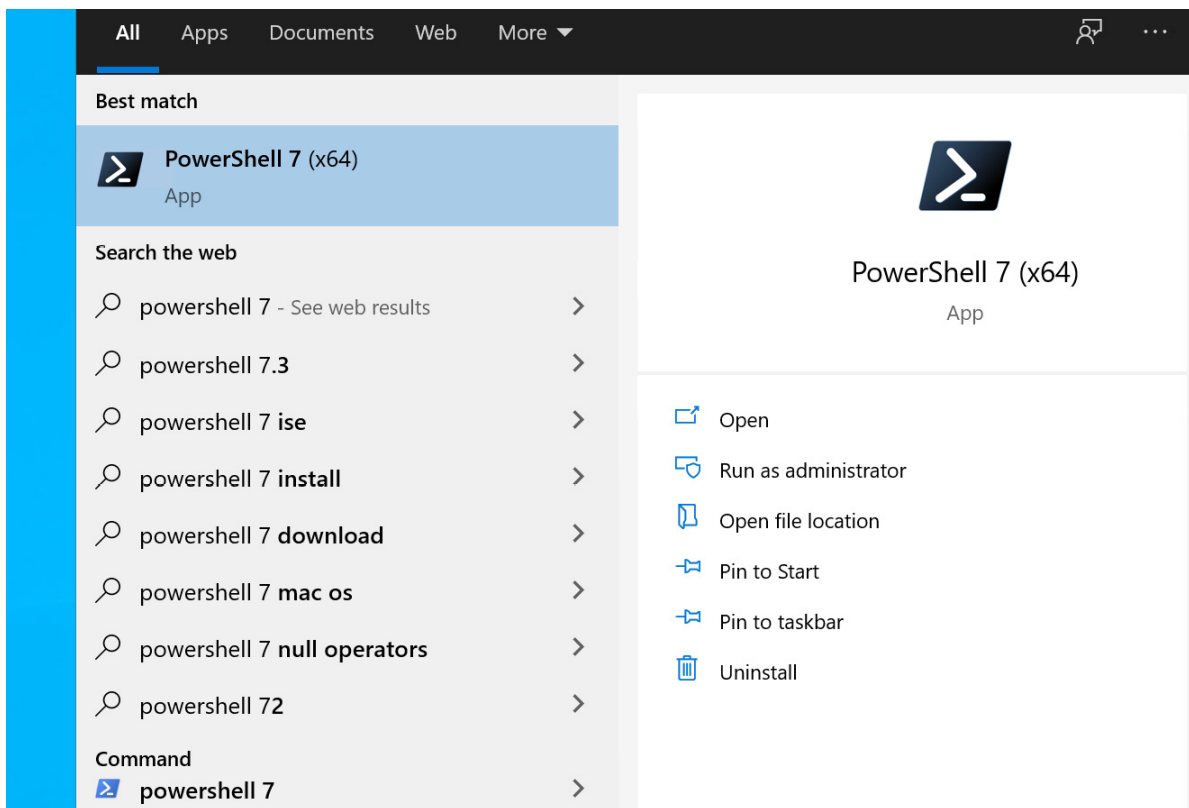


Figure 6.15 – Searching for PowerShell

2. Check the version of PowerShell by running the following command from within PowerShell:

```
$PSVersionTable.PSVersion
```

The output will look as follows:

```
Administrator: PowerShell 7 (x64)
PowerShell 7.1.3
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Users\vmadmin> $PSVersionTable.PSVersion

Major  Minor  Patch  PreReleaseLabel  BuildLabel
-----
7      1      3
PS C:\Users\vmadmin> _
```

Figure 6.16 – PowerShell version

3. Enter the following command; the PowerShell script execution policy must be set to *remote signed* or *less restrictive*:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```

Confirm this using the following command:

```
Get-ExecutionPolicy -List
```

The output will be as follows:

```
Administrator: PowerShell 7 (x64)
PS C:\Users\vmadmin> Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
PS C:\Users\vmadmin>
>>
PS C:\Users\vmadmin> Get-ExecutionPolicy -List

Scope ExecutionPolicy
-----
MachinePolicy      Undefined
UserPolicy         Undefined
Process            Undefined
CurrentUser        RemoteSigned
LocalMachine       RemoteSigned
```

Figure 6.17 – Setting the execution policy

4. Enter the following command to install the Az module:

```
Install-Module -Name Az -AllowClobber -Scope CurrentUser -Repository
PSGallery -Force
```


In this exercise, we looked at installing the Azure Powershell module. We will install the Azure CLI in the following exercise.

Exercise 2 – installing the Azure CLI

In this section, you'll learn how to install the Azure CLI.

The following steps must be carried out on the OS of a machine you have admin access to; this could be physical or virtual. We are using a Windows 10 device for this exercise – *Windows 10 Pro 20H2* using *PowerShell 7* for reference. Once installed, the CLI can be accessed via *PowerShell* or the **Windows Command Prompt (CMD)**:

1. From your device, search for **Windows PowerShell 7** and click **Open**.

Enter the following command:

```
Invoke-WebRequest -Uri https://aka.ms/installazurecliwindows -OutFile  
.AzureCLI.msi; Start-Process msixexec.exe -Wait -ArgumentList '/I  
AzureCLI.msi /quiet'; rm .\AzureCLI.msi
```

The following screenshot shows the installation progress:

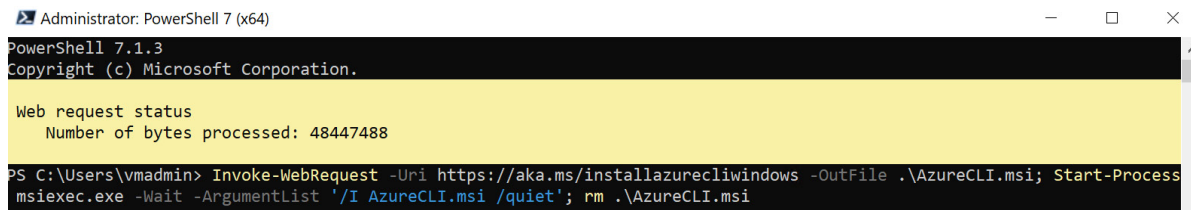


Figure 6.20 – Installing the Azure CLI via PowerShell

2. Close PowerShell and reopen it, Then, enter the following command to run the CLI from PowerShell:
az login
3. When the pop-up Azure sign-in page appears, as shown in the following screenshot, sign in:

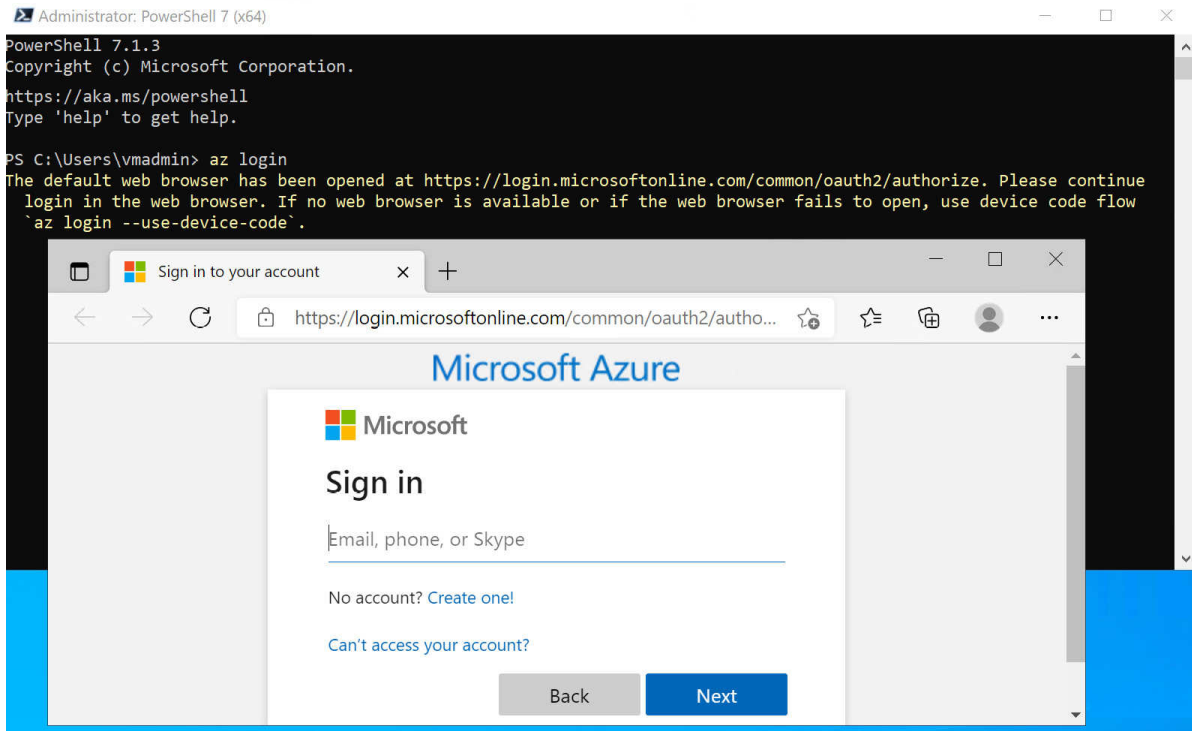


Figure 6.21 – Azure Sign in page

4. Enter the following command to check the version of the CLI that's been installed:

```
az version
```

The output will look as follows:

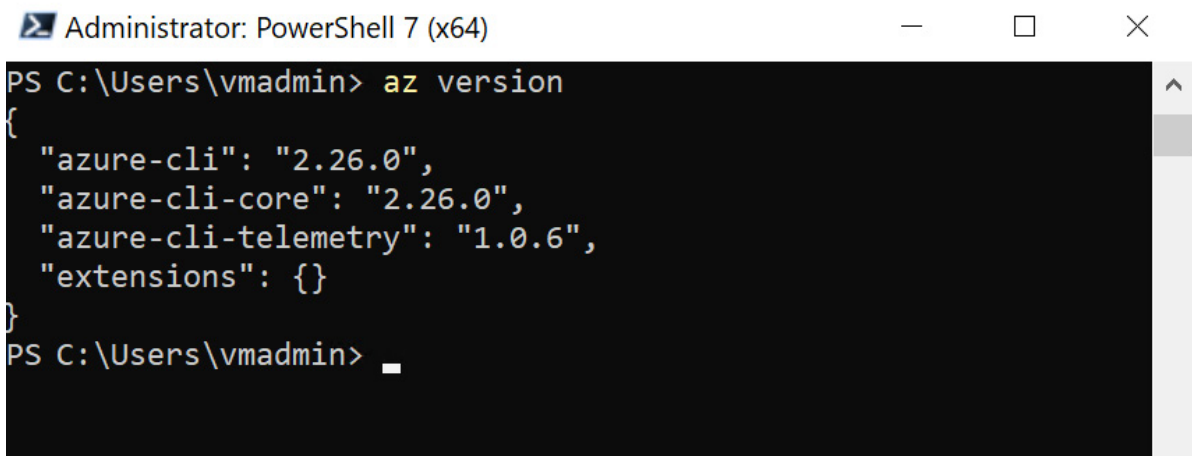
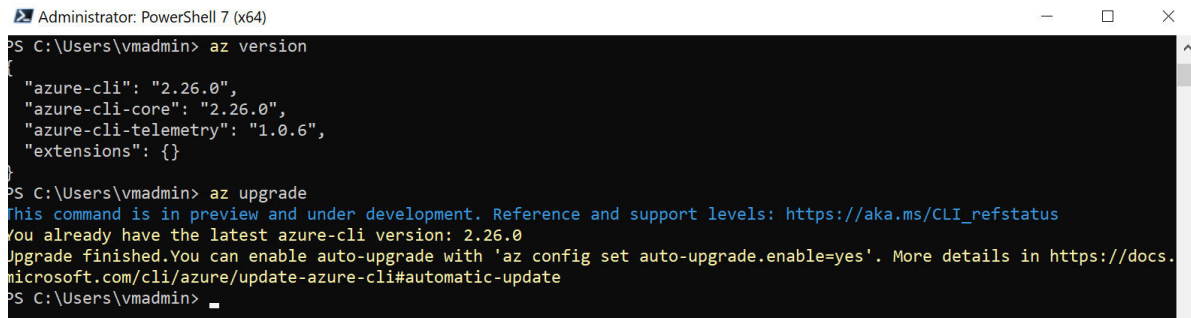


Figure 6.22 – CLI version

5. Enter the following command to update to the latest version:

```
az upgrade
```

6. The output will look as follows:



```
Administrator: PowerShell 7 (x64)
PS C:\Users\vmadmin> az version
{
  "azure-cli": "2.26.0",
  "azure-cli-core": "2.26.0",
  "azure-cli-telemetry": "1.0.6",
  "extensions": {}
}

PS C:\Users\vmadmin> az upgrade
This command is in preview and under development. Reference and support levels: https://aka.ms/CLI_refstatus
You already have the latest azure-cli version: 2.26.0
Upgrade finished.You can enable auto-upgrade with 'az config set auto-upgrade.enable=yes'. More details in https://docs.microsoft.com/cli/azure/update-azure-cli#automatic-update
PS C:\Users\vmadmin>
```

Figure 6.23 – CLI upgrade

7. The command's output in the preceding screenshot shows that the latest updates have been installed and that no is upgrade available; this final command concludes this exercise.

We looked at installing the Azure CLI in this exercise. In the following exercise, we will create a resource group and a virtual machine using PowerShell from Cloud Shell.

Exercise 3 – creating resources using PowerShell from Cloud Shell

In this section, we'll learn how to interact with Azure environments using PowerShell as the CLI tool and Cloud Shell as the shell environment for running commands.

The following subsections will show you how to create the necessary resources. These have been segregated into tasks for ease of understanding:

Task – accessing the Azure portal

1. Log in to the Azure portal by going to <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – setting up Cloud Shell (skip this task if you've already completed it)

2. Click on the Cloud Shell `code` icon on the top toolbar of the portal; if this is the first time you have used Cloud Shell and no storage has been mounted to persist files, you will be prompted to create a storage account. Select your subscription and then click on **Create storage**:

The screenshot shows the Microsoft Azure portal dashboard. The top navigation bar includes the Microsoft Azure logo, a search bar, and several utility icons. A red box highlights the Cloud Shell icon (a terminal window with a code symbol). Below the navigation bar is the 'My Dashboard' section, which includes a search bar, a refresh button, and various dashboard controls. The main content area displays several tiles: 'Azure getting started made easy!', 'All resources' (Test), 'Marketplace', 'milesbetter.solutions', 'Users and groups', 'Resource groups' (Test), and 'Help + support'. At the bottom of the screenshot, a modal dialog is open with the title 'You have no storage mounted'. The dialog contains the text: 'Azure Cloud Shell requires an Azure file share to persist files. [Learn more](#). This will create a new storage account for you and this will incur a small monthly cost. [View pricing](#)'. Below this text is a field for 'Subscription' with the value 'Project Lobster' and a 'Show advanced settings' link. At the bottom of the dialog are two buttons: 'Create storage' and 'Close'.

Figure 6.24 – Cloud Shell setup

3. Once the storage account has been created, you will be connected to a terminal running the shell environment you will use to run commands.

Task – creating a resource group

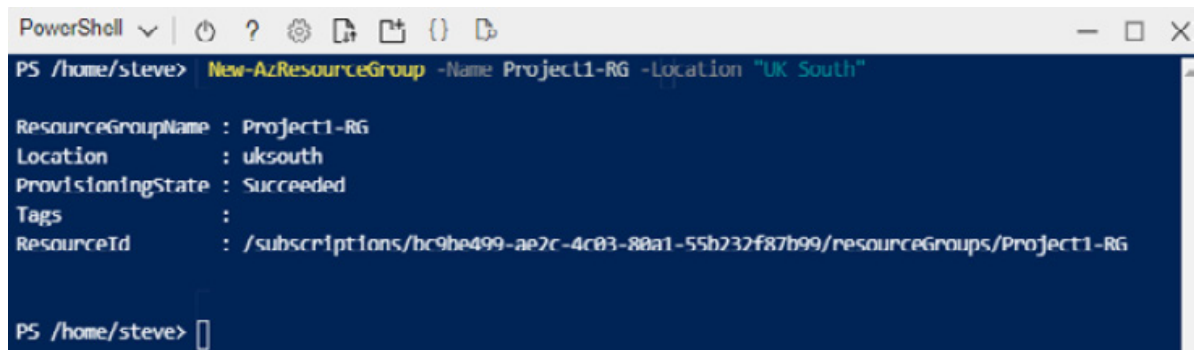
4. *PowerShell* should be set as the CLI within Cloud Shell for this exercise.
5. In this exercise, we will use an example location; you may use a location that meets your needs. You can use the following command to find a location to use to create resources in:

Get-AzLocation

6. To create a resource group, enter the following command:

New-AzResourceGroup -Name Project1-RG -Location "UK South"

The output will look as follows:



```
PowerShell | PS /home/steve> New-AzResourceGroup -Name Project1-RG -Location "UK South"
ResourceGroupName : Project1-RG
Location           : uksouth
ProvisioningState  : Succeeded
Tags               :
ResourceId         : /subscriptions/bc9be499-ae2c-4c83-88a1-55b232f87b99/resourceGroups/Project1-RG
PS /home/steve> 
```

Figure 6.25 – Creating a resource group

7. To show the resource group that has just been created, enter **Get-AzResourceGroup** or **Get-AzResourceGroup | Format-Table**.
8. The created resource group will also appear in the portal UI.

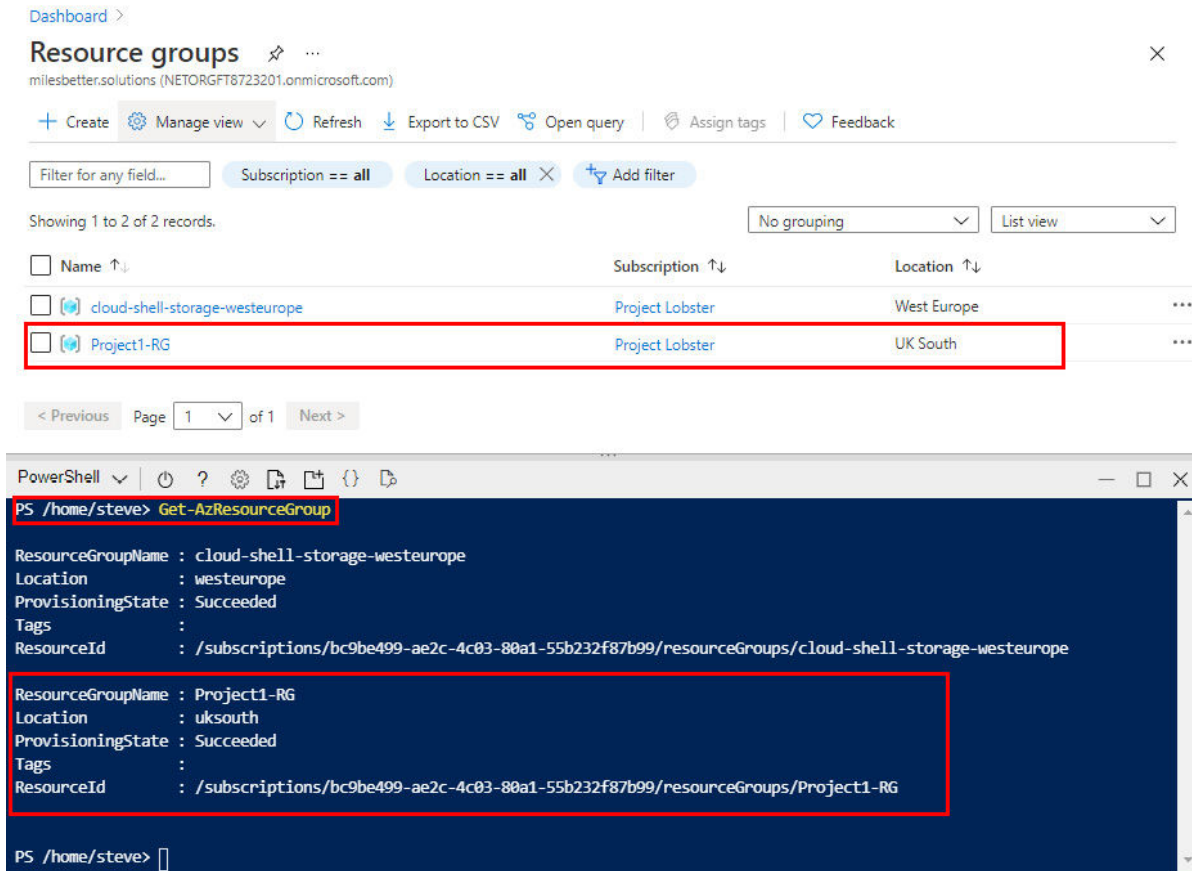


Figure 6.26 – Showing resource groups

Task – creating a virtual machine

9. To create a virtual machine, enter the following code:

```

New-AzVm `
  -ResourceGroupName "myResourceGroup" `
  -Name "MyVM" `
  -Location "UK South" `
  -VirtualNetworkName "myVnet" `
  -SubnetName "mySubnet" `
  -SecurityGroupName "myNetworkSecurityGroup" `
  
```

10. When prompted, enter a username and password.

```
PowerShell | PS /home/steve> New-AzVm
>> -ResourceGroupName "myResourceGroup"
>> -Name "MyVM"
>> -Location "UK South"
>> -VirtualNetworkName "myVnet"
>> -SubnetName "mySubnet"
>> -SecurityGroupName "myNetworkSecurityGroup"
>>

cmdlet New-AzVM at command pipeline position 1
Supply values for the following parameters:
Credential
User: vmuser
Password for user vmuser: *****]
```

Figure 6.27 – Creating a virtual machine

11. You will see a progress message while the virtual machine is being created.

```
PowerShell | cmdlet New-AzVM at command pipeline position 1
Supply values for the following parameters:
Credential
User: vmuser
Password for user vmuser: *****

No Size value has been provided. The VM will be created with the default size Standard_D2s_v3.
[
  Creating Azure resources
  18% \
  [ooooooooooooooooooooooooooooo]
  Creating virtualMachines/MyVM. ]
```

Figure 6.28 – Virtual machine creation progress

12. Once the virtual machine has been created, you will see a **ProvisioningState** of **Succeeded**.

```
PowerShell | [Icons]
Tags : {}
HardwareProfile : {VmSize}
NetworkProfile : {NetworkInterfaces}
OsProfile : {ComputerName, AdminUsername, WindowsConfiguration, Secrets, AllowExtensionOperations, RequireGuestProvisionSignal}
ProvisioningState : Succeeded
StorageProfile : [ImageReference, OsDisk, DataDisks]
FullyQualifiedDomainName : myvm-8c333a.UK South.cloudapp.azure.com
_
PS /home/steve>
```

Figure 6.29 – Virtual machine created

13. To show the virtual machine that was just created, enter the following command:

```
get-AzVM
```

14. The created virtual machine will also appear in the portal UI.

The screenshot shows the Azure portal interface for 'Virtual machines'. It displays a table with the following data:

Name	Subscription	Resource group	Location	Status	Operating system	Size
MyVM	Project Lobster	myResourceGroup	UK South	Running	Windows	Standard_D2s_v3

Below the screenshot, a PowerShell terminal window shows the output of the `get-AzVM` command:

```

ResourceGroupName Name Location VmSize OsType NIC ProvisioningState Zone
-----
MYRESOURCEGROUP MyVM uksouth Standard_D2s_v3 Windows MyVM Succeeded
  
```

Figure 6.30 – Showing the created virtual machine

In this exercise, we looked at creating a resource group and a virtual machine using PowerShell via Cloud Shell. In the following exercise, we will create a resource group and a virtual machine using the Azure CLI from Cloud Shell.

Exercise 4 – creating resources using the Azure CLI from Cloud Shell

In this exercise, you'll learn how to interact with Azure environments using the Azure CLI as the CLI tool and Cloud Shell as the shell environment for running commands.

The following subsections will show you how to create all the necessary resources for this exercise. These have been segregated into tasks for ease of understanding:

Task – accessing the Azure portal

1. Log in to the Azure portal by going to <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – setting up Cloud Shell (skip this task if you've already completed it)

2. Click on the Cloud Shell *code* icon on the top toolbar of the portal. If this is the first time you have used Cloud Shell and no storage has been mounted to persist files, you will be prompted to create a storage account. Select your subscription and then click on **Create storage**:

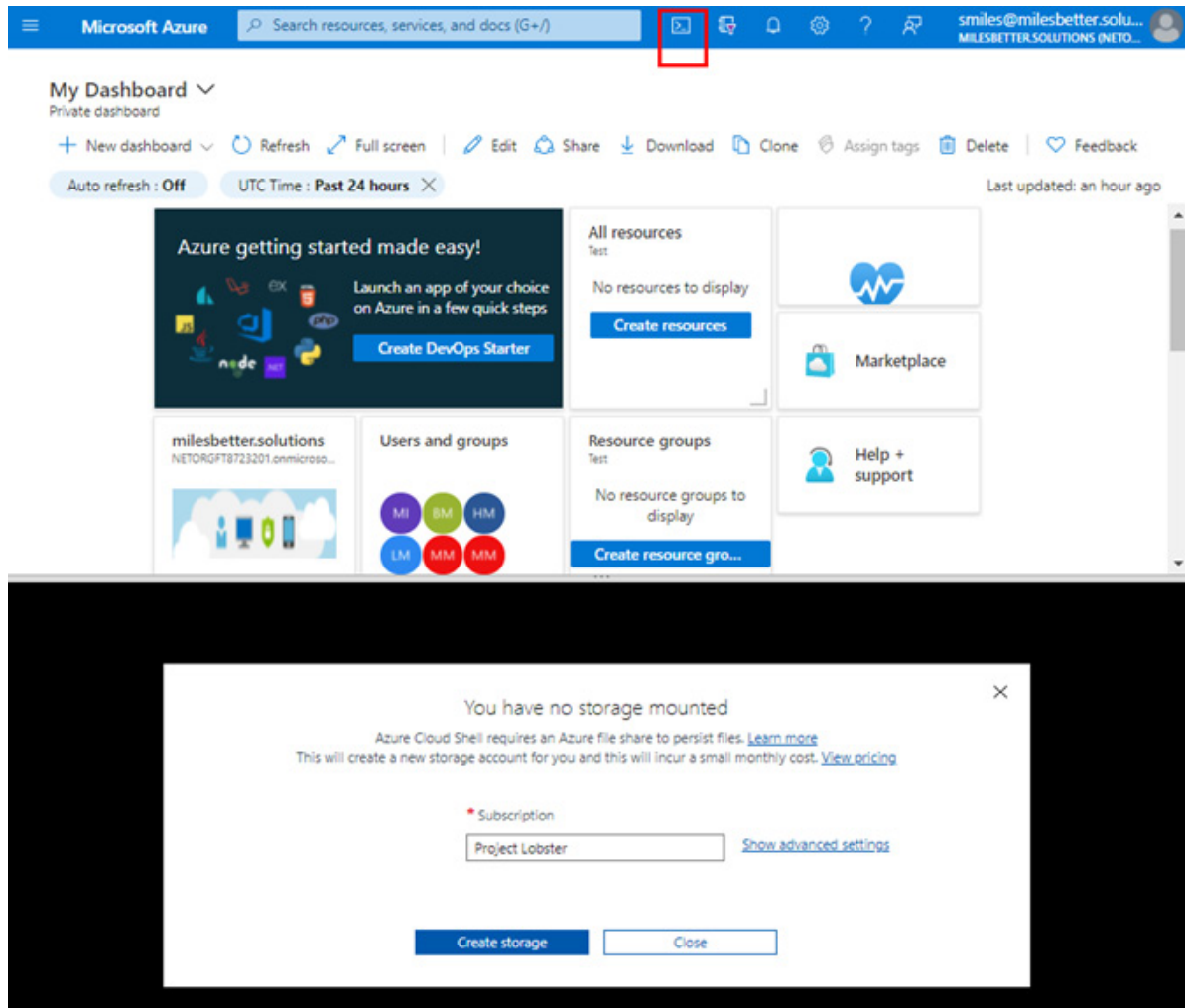


Figure 6.31 – Cloud Shell setup

- Once the storage account has been created, you will be connected to a terminal running the shell environment you will use to run commands.

Task – creating a resource group

- Bash* should be set as the CLI within Cloud Shell for this exercise.
- To create a resource group, enter the following command:

```
az group create -l uksouth -n Project1-RG
```

- The output will look as follows:

```
Bash
steve@Azure:~$ az group create -l uksouth -n Project1-RG
{
  "id": "/subscriptions/hc9he499-ae7c-4c83-88a1-55h73df87h99/resourceGroups/Project1-RG",
  "location": "uksouth",
  "managedby": null,
  "name": "Project1-RG",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
steve@Azure:~$
```

Figure 6.32 – Creating a resource group

7. To show the resource group that has just been created, enter `az group list` or `az group list --out table`.
8. The created resource group will also appear in the portal UI.

The screenshot shows the Azure portal interface for 'Resource groups'. It displays a table with three records. The 'Project1-RG' record is highlighted with a red box. Below the portal, a terminal window shows the command `az group list --out table` and its output, which lists three resource groups: 'cloud-shell-storage-westeuropa', 'Project1-RG', and 'NetworkWatcherRG'. The 'Project1-RG' row in the terminal output is also highlighted with a red box.

Name	Subscription	Location
cloud-shell-storage-westeuropa	Project Lobster	West Europe
NetworkWatcherRG	Project Lobster	UK South
Project1-RG	Project Lobster	UK South

Name	Location	Status
cloud-shell-storage-westeuropa	westeuropa	Succeeded
Project1-RG	uksouth	Succeeded
NetworkWatcherRG	uksouth	Succeeded

Figure 6.33 – Showing the created resource group

Task – creating a virtual machine

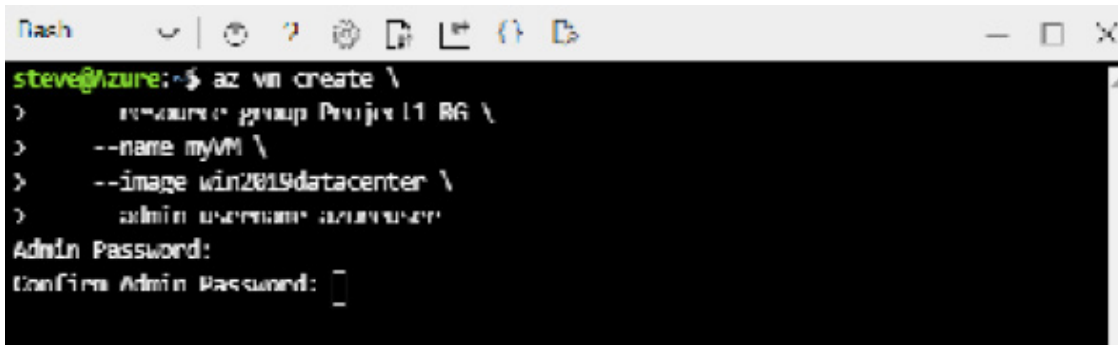
9. To create a virtual machine, enter the following code:

```
az vm create \
  --resource-group Project1-RG \
```



```
--name myVM \  
  
--image win2019datacenter \  
  
--admin-username azureuser
```

10. When prompted, enter a username and password:

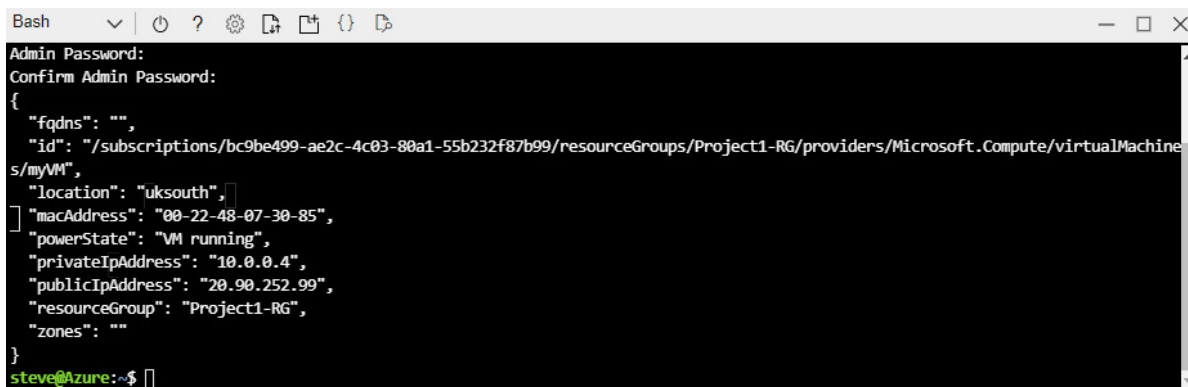


```
Bash  
steve@Azure:~$ az vm create \  
> --resource-group Project1-RG \  
> --name myVM \  
> --image win2019datacenter \  
> --admin-username azureuser  
Admin Password:  
Confirm Admin Password: [ ]
```

Figure 6.34 – Creating a virtual machine

11. You will see a **Running** message while the virtual machine is being created.

12. Once the virtual machine has been created, the following information will be returned:



```
Bash  
Admin Password:  
Confirm Admin Password:  
{  
  "fqdns": "",  
  "id": "/subscriptions/bc9be499-ae2c-4c03-80a1-55b232f87b99/resourceGroups/Project1-RG/providers/Microsoft.Compute/virtualMachines/myVM",  
  "location": "uksouth",  
  "macAddress": "00-22-48-07-30-85",  
  "powerState": "VM running",  
  "privateIpAddress": "10.0.0.4",  
  "publicIpAddress": "20.90.252.99",  
  "resourceGroup": "Project1-RG",  
  "zones": ""  
}  
steve@Azure:~$ [ ]
```

Figure 6.35 – Virtual machine created

13. To show the virtual machine that has just been created, enter the following command:

```
az vm list --out table
```

14. The created virtual machine will also appear in the portal UI.

Dashboard >

Virtual machines ✎ ...

milesbetter.solutions (NETORGFT8723201.onmicrosoft.com)

+ Create ↻ Switch to classic 🕒 Reservations ⚙️ Manage view 🔄 Refresh 📄 Export to CSV 🔗 Open query 🏷️ Assign tags ...

Subscription == all
Resource group == all
Location == all
+ Add filter

Showing 1 to 1 of 1 records. No grouping List view

<input type="checkbox"/> Name ↑↓	Subscription ↑↓	Resource group ↑↓	Location ↑↓	Status ↑↓	Operating system ↑↓	Size ↑↓
<input type="checkbox"/> myVM	Project Lobster	Project1-RG	UK South	Running	Windows	Standard

< Previous Page 1 of 1 Next >

```

Bash
steve@Azure:~$ az vm list --out table
Name      ResourceGroup  Location  Zones
-----
myVM      PROJECT1-RG    uksouth
steve@Azure:~$

```

Figure 6.36 – Showing the created virtual machine

In this exercise, we created a resource group and a virtual machine using the Azure CLI from Cloud Shell. The following exercise looks at using Azure Service Health in the Azure portal.

Exercise 5 – exploring Azure Service Health

In this section, you'll learn how to interact with Azure Service Health in the Azure portal:

Task – accessing the Azure portal

1. Log in to the Azure portal by going to <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – exploring Azure Service Health

2. From the search box at the top of the portal, enter **Service Health** and click on **Service Health** from the results.
3. On the **Service Health** page, there is a left-hand navigation menu. Here, you can view **ACTIVE EVENTS, HISTORY, RESOURCE HEALTH, and ALERTS**.
4. From **Service issues**, under the **ACTIVE EVENTS** section, you can launch a *guided tour* and view the current service issues or issues that have been resolved in the past 7 days that may be impacting your resources; the map will show color status dots for each region you have resources in. This can be seen in the following screenshot:

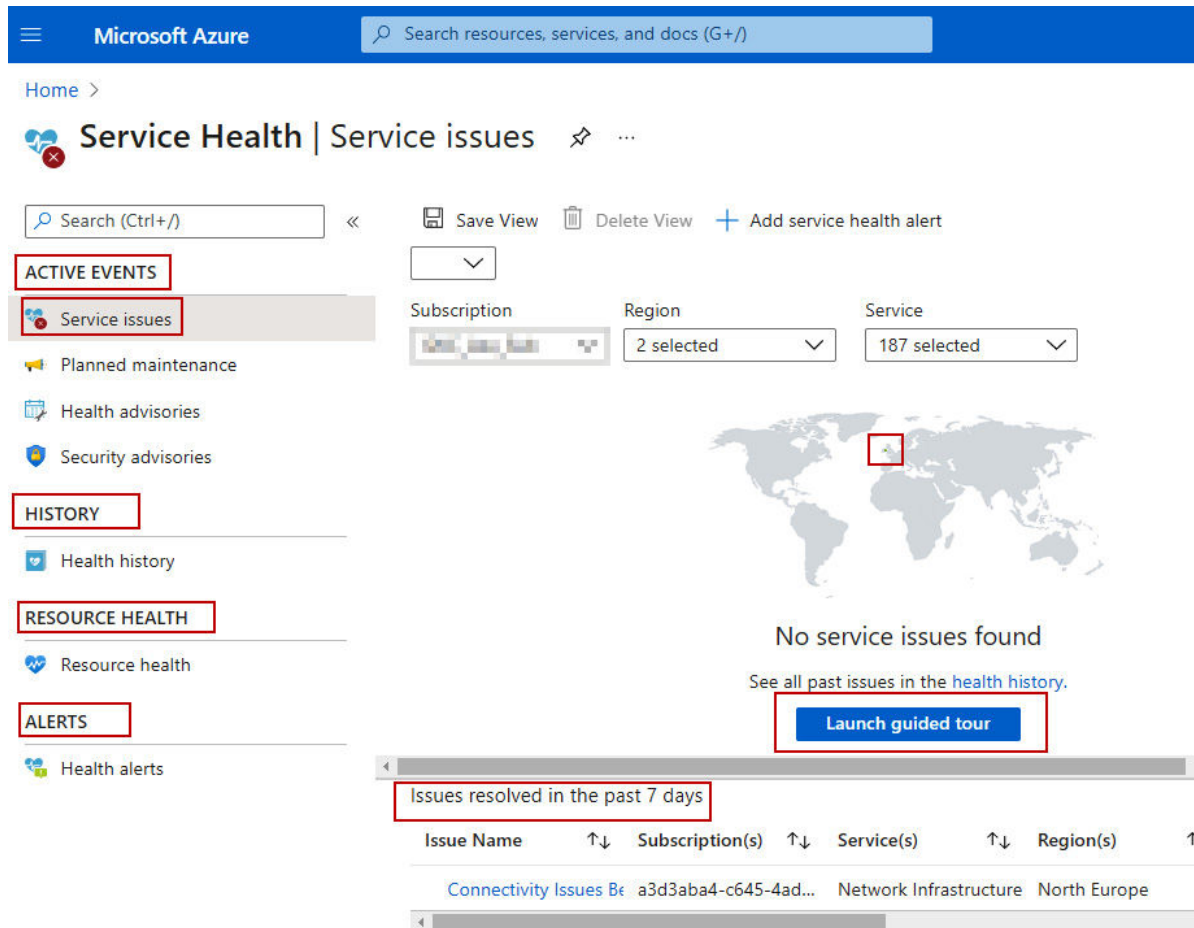


Figure 6.37 – Service Health

5. In the **ACTIVE EVENTS** section, you can also click through to see any **Planned maintenance** events that may impact your resources, **Health advisories**, and **Security advisories**; for each, there is a click-through link to see all past issues in the **Health history** section
6. In the **Health History** section, you can click on a service issue, find out more information about the impact, any updates, and download a summary report and a **Root Cause Analysis (RCA)** report.
7. In the **Resource health** section, you can view the health of any individual resource you have created within your subscription(s) so that you know if everything is running as expected. If any problems are impacting its running, you will be told and actions you can take will be provided. For example, if a VPN gateway is not running normally, a hyperlink will be provided so that you can reset it.
8. In the **Health alerts** section, you can add a service health alert rule so that a notification is sent based on a set of service health criteria.

With that, we've covered the hands-on exercises for this chapter. Now, let's summarize what we've learned.

Summary

This chapter provided complete coverage of the AZ-900 Azure Fundamentals exam skills section called *Describing management tools on Azure*.

In this chapter, you learned about various skills that will provide you with the confidence to explain and discuss the functionality and usage of the following aspects with a business or technical audience: the Azure portal, Azure PowerShell, the Azure CLI, Cloud Shell, the Azure mobile app, Azure Advisor, Azure Monitor, and Azure Service Health.

Further knowledge beyond the exam objectives was provided to help you prepare for a real-world, day-to-day, Azure-focused role.

We concluded this chapter with a hands-on exercise section that brought together some of the skills areas that were covered in this chapter.

The next chapter will outline the general security features that are available in Azure, including Azure Security Center, Azure Sentinel and Defender, Azure Key Vault, Azure Dedicated Host, and Azure Network Security.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure fundamentals:
<https://docs.microsoft.com/learn/certifications/exams/az-900>
- Exam AZ-900: Skills outline: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- Microsoft Learn: Azure Fundamentals – Describe core solutions on Azure:
<https://docs.microsoft.com/learn/paths/az-900-describe-core-solutions-management-tools-azure>

Skill check

Challenge yourself with what you have learned in this chapter by answering the following questions:

1. Explain the Azure portal and how it can be accessed.
2. How is governance implemented and access controlled within the portal?
3. What is Azure PowerShell, and when can it be used?
4. What is the Azure CLI, and when can it be used?
5. How does the Azure CLI differ from PowerShell?
6. What is Azure Cloud Shell, and how does it relate to PowerShell and the Azure CLI?
7. What is the Azure mobile app, and what devices does it support?
8. What is Azure Advisor? Are the recommendations mandatory to implement?
9. What is Azure Monitor?
10. What is Azure Health Service and how does it relate to the Azure Status page?

Section 4: Security

This section provides complete coverage of the knowledge and skills required for the *skills measured* of the *Describe general security and network security features* section of the exam. We also cover knowledge and skills that go beyond the exam content, so you are prepared for a real-world, day-to-day Azure-focused role.

This part of the book comprises the following chapter:

- [Chapter 7](#), *Azure Security*

Chapter 7: Azure Security

In [Chapter 6](#), *Azure Management Tools*, you learned the skills that covered Azure Advisor, Azure Monitor, Azure Service Health, the Azure portal, Azure PowerShell, and the **Azure command-line interface (Azure CLI)**.

This chapter will outline the security aspects available in Azure, including security concepts and the security services themselves that you can enable, as well as security posture management and security operations tooling.

This chapter aims to provide complete coverage of the *AZ-900 Azure Fundamentals – Skills Measured* section: *describe general security and network security features*.

By the end of this chapter, you will have learned the following skills:

- Describe the concept of threat modeling.
- Describe the concept of Zero Trust.
- Describe the concept of **defense in depth (DiD)**.
- Describe the functionality and usage of Azure Key Vault and Azure Dedicated Host.
- Describe the functionality and usage of **network security groups (NSGs)**, Azure Firewall, and Azure DDoS Protection.
- Describe the functionality and usage of Azure Security Center and Azure Sentinel.

To support your learning with some practical skills, we will also look at the hands-on usage of some of the tools covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – Create an Azure key vault.
- Exercise 2 – Secure network access using an NSG.

This chapter's goal is also to take your knowledge beyond the exam objectives so that you are prepared for a real-world, day-to-day Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will require the following:

- An Azure subscription that has access to create and delete resources in the subscription. If you do not have an Azure subscription, you can create a free Azure account from this link: <https://azure.microsoft.com/free>.
- Access to an internet browser, as you will log in to the Azure portal at <https://portal.azure.com>.
- You can alternatively use the Azure desktop app, available at <https://portal.azure.com/App/Download>.

It is important to note that in November 21 some Microsoft Security Services have been renamed. These are renamed as follows:

- **Azure Security Center** and **Azure Defender** are now called **Microsoft Defender for Cloud**
- **Azure Defender plans** to **Microsoft Defender plans**
- **Azure Sentinel** is now called **Microsoft Sentinel**
- **Azure Defender for IoT** is now called **Microsoft Defender for IoT**
- **Azure Defender for SQL** is now called **Microsoft Defender for SQL**
- **Microsoft Cloud App Security** is now called **Microsoft Defender for Cloud App**
- **Microsoft Defender for Business** is introduced as a new Service SKU

You can learn more about the update of Microsoft security services at this url:

<https://docs.microsoft.com/en-us/azure/defender-for-cloud/defender-for-cloud-introduction>

Threat modeling

Attackers can take many forms, such as *criminal hackers*, *hacktivists*, *competitors*, and *foreign nations*. Don't forget either that attackers are not only external; they can be *internal* to an organization—for example, *ex-employees*—these often being the hardest to detect and prevent. For further reading, you should enter *sly Dog gang* into your favorite search engine to read about a real-world insider espionage attack on one of the highest-profile manufacturers of electric vehicles.

You must put in measures so that you don't become an easy target for opportunists as well as the crafted, pre-meditated, military-style operation of some sophisticated attacks; these measures are designed to raise the attacker's costs significantly, so they divert their resources and activities to an easier attack target that has a higher return on their attack investment.

The approach that should be taken is to adopt a **threat priority model**; this can then aid in identifying your threat priorities and where security investments should be made to reduce your costs of security operations and increase your attacker's kill-chain costs. The following diagram aims to visualize this approach:

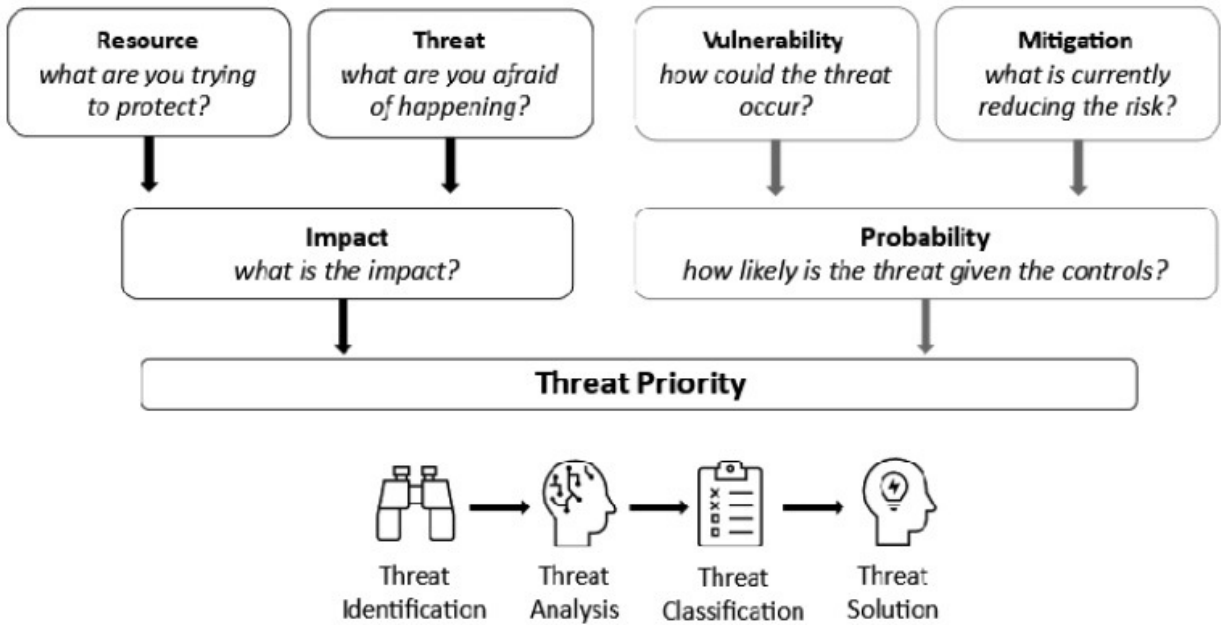


Figure 7.1 – Threat priority model

Any security approach must start from an inward look at your current security position and secure score. A *secure score* can be thought of like a credit-rating score you receive to see how likely you are to be accepted for a finance agreement, but in security terms, it looks at where you are on the attack vulnerability scale of 1 to 10, as it were; this score will indicate your security posture.

A **security posture** is an organization's threat-protection and response capabilities; this ensures that an organization has the ability for systems, data, and identities to be recoverable and operational should an attack be successful. It is critical to understand that we cannot prevent or eliminate threats and attacks, and the fact is that an attacker only has to be successful once while you must protect everything, all the time. A security posture's goal should be to reduce exposure to threats, shrinking attack surface areas and vectors while building resilience to attacks, as they cannot be eliminated.

A security strategy and security posture should use the guiding principles of *Confidentiality, Integrity, and Availability*, also referred to as the **CIA triangle**. There is no perfect threat prevention or security solution; there will always be a trade-off, and the CIA model is a way to think about that. The CIA model is a common industry model used by security professionals; it is not a Microsoft model. Let's look at these guiding principles in more detail here:

- **Confidentiality**—This is a requirement that sensitive data is kept protected and can only be accessed by those who should have access through the **principle of least privilege (POLP)**. Confidentiality is about the confidence that the data cannot be accessed, read, or interpreted by anybody other than those intended to read and access this data; this can be achieved by *encrypting* the data. The encryption keys also need to be made confidential and available to those who need access to the data.
- **Integrity**—This means that data transferred is the same as data received; the bytes sent are the same bytes received. Integrity is about the confidence that the data has not been altered from its original form or tampered with; this can be achieved by *hashing the data*. Malware can threaten the integrity of systems and data.
- **Availability**—This means that data and systems are available to those that need them, including access to encryption keys, but in a secure and governed manner. Availability means a trade-off between the three sides of the triangle and a balance being made of being locked down

for security but accessible for operational needs and productivity. A **distributed-denial-of-service (DDoS)** attack will threaten the availability of systems, data, and encryption keys.

The following screenshot represents the CIA triangle model:



Figure 7.2 – Security posture CIA triangle

The aim of an attack may be specific to an organization and may be different based on the form of the attacker—such as a criminal hacker, a foreign nation, a hacktivist, an opportunist, and so on. The aim may be to steal data, deface a website, alter the integrity of an app or a service, extort money through ransom, and so on.

There are two motivations of attackers, *money* or *mission*. The motivation is clearer for money-driven attacks and has a certain level of calculation by the attacker on their **return on investment (ROI)** before they give up and move on to another target. However, for mission-driven attacks, the rationale may be more opaque and less tangible of what is to be gained, and a mission attack is often more of a moral standard and a matter of ethics, principles, politics, and control than money. Thus, the attacks may be more sustained and the attackers determined to succeed at any cost, because the reward may not have a price that can be attached. The following diagram aims to visualize this approach:

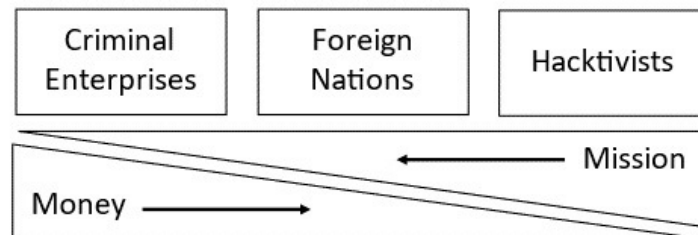


Figure 7.3 – Attack motivations

We have learned about the types of attackers and their motivations; the following are some of the most common threats to protect against:

- **Ransomware**—This is malware that will encrypt files and folders in an attempt to extort money.
- **Data breach**—This includes phishing, spear phishing, **Structured Query Language (SQL)** injection, stealing passwords/bank details/other sensitive information, luring somebody to click a link, and opening a file.
- **Dictionary attack**—This is an identity-theft attack, also known as a brute-force attack; known passwords are used against an account to steal an identity.
- **Disruptive attack**—This is a network and workload attack; a DDoS attack attempts to make a network or workload unavailable by flooding it with requests and attempting to exhaust its resources.

Attackers plan and structure their attacks; this is so they can live undetected on the network and in the user's systems without the victim being alerted. As the adage says, there are two types of organizations: *those who have been compromised and those who don't know yet.*

Attacks follow a *sequence* or *chain* of events; this is known as an **attack chain** or a **kill chain**. The following diagram shows a common chain:

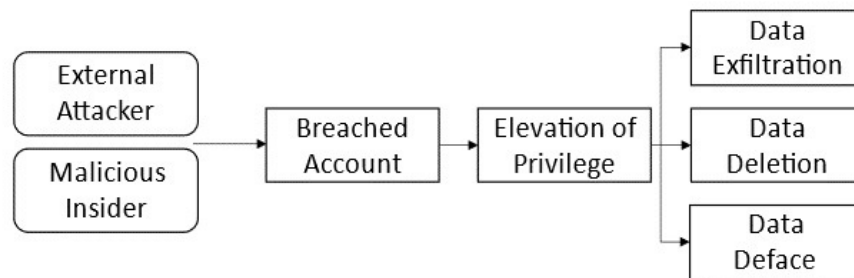


Figure 7.4 – Attack chain

When a user account is compromised, it can access the network and then work to elevate privileges to an admin account that can then move laterally within the network to access the data and execute activities such as steal, delete, corrupt, and encrypt data.

Through a *Zero Trust* and *DiD* approach to protecting assets, the goal is to prevent and disrupt this chain of events; we want to put multiple obstacles in the attacker's

way and increase their attack costs so that they will move on to launching an easier attack elsewhere that offers less resistance.

Security can often be seen as the *anti-pattern* of *operations*, *availability*, and *productivity*; you may have encountered overzealous security teams referred to as *business prevention teams*. Much as there have been silos and cultural divides between *development* and *operations* teams, there is often a divide between *security* and these teams.

Often, the feeling is that it's the security team's job to make things secure and protect code, data, systems—a *not my job* attitude, throwing it over the wall in an *it's the security team's problem now* culture.

Security must be in place before a single line of code is written, a system created, or data stored; a culture akin to **Development-Operations (DevOps)** of fostering trust between all teams and security teams must exist, and leaders must bring the notion and culture of **Security-Development-Operations (SecDevOps)** into an organization.

The bottom line is that security is not just *somebody else's problem*, but *everybody's responsibility*; and as they say... *if you are not part of the solution, you are part of the problem*.

Zero Trust

An important concept to consider is **Zero Trust**, which uses the approach of *never trust, always verify*; this concept relates to thinking beyond traditional network perimeter-based security and adopting a holistic approach to security.

Zero Trust is not a service or solution but a wider-thinking security strategy and framework to be adopted, and works on the notion of ensuring compliance and securing access at the resource and no longer the location or network the resource is on; we must *NOT assume trust* because of the resource's network or location.

The Zero Trust framework is built upon the following foundational principles:

- Assume breach.
- Verify explicitly.
- Use least-privilege access (**Just In Time (JIT)**, and just enough access).

In this new world of hybrid work where organizations' traditional firewalls and security service-controlled network perimeters have vanished due to remote working, we must now consider *identity* as the new perimeter. The following are the Zero Trust framework's six foundational elements:

- **Identities**—Users, services, devices; each represents an element to be compromised
- **Devices**—Represent an attack surface and threat vector for data flows
- **Applications**—Represent the consumer of the data flows
- **Data**—Represents the data stored that is to be protected
- **Infrastructure**—Represents an attack surface and threat vector, whether locally on-premises or remotely hosted by a cloud provider
- **Network**—Represents an attack surface and threat vector and should be segmented

In this section, we introduced the concept of the Zero Trust framework. The following section looks at the concept of DiD.

Defense in depth

DiD refers to a strategy that places multiple layers of different forms of defenses between attackers and the resources you are trying to protect.

Adopting a DiD strategy allows an organization to adopt a strong security posture and help ensure that all systems, data, and users are better protected from threats and compromise.

A DiD strategy means there is no single layer of protection or security service that is solely responsible for protecting resources, but by implementing many different types of defense at individual layers, you can slow down an attack path. It may successfully breach one defensive layer but be halted by subsequent protection layers, preventing the protected resource from being exposed.

The following screenshot shows that DiD as a concept is nothing new as a strategy; it can be considered the *medieval castle* concept of protecting resources:

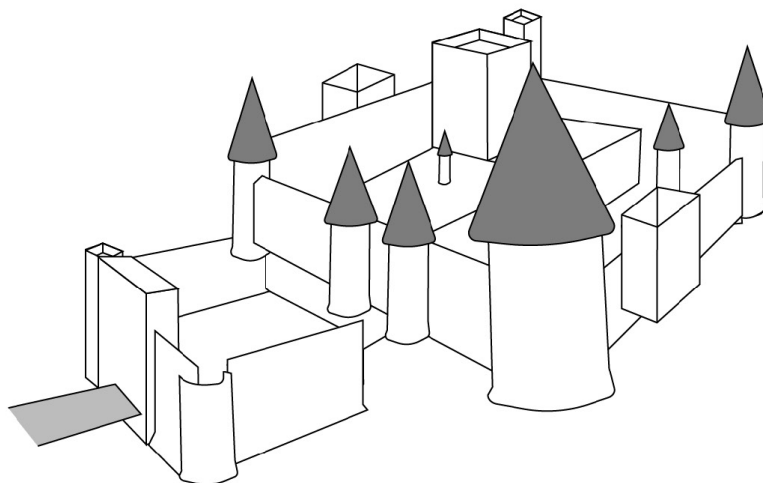


Figure 7.5 – Medieval castle defense approach

The medieval castle approach should be part of your strategy for building your resources in Azure; you define multiple layers that can be protected by different security services that are the most appropriate at each layer.

As with our medieval castle analogy, each layer from the center to the outside to the center provides its own independent protection service, tailored to best protect the characteristics of that layer.

The following diagram aids in visualizing the layers that make up a DiD strategy for a resource to be protected:

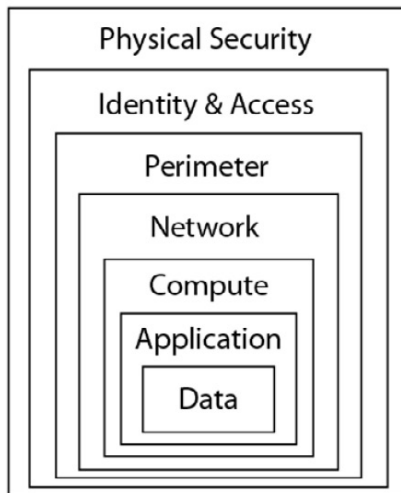


Figure 7.6 – DiD approach layers

There is no one-size-fits-all security service that can protect all the layers; however, we must have security services at each layer that work in conjunction and complement the layers outside and inside of their layer. There must be a single unified view so that telemetry and threat intelligence can be passed between each layer and enhance the protection at each layer. Microsoft uses **artificial intelligence (AI)**, *threat intelligence*, and *analytics* to enhance these capabilities.

In this section, we looked at adopting a DiD strategy. The following section looks at network and application protection.

Network and application protection

This section introduces the core solutions available in Azure to protect and secure the network and applications running in Azure; this section also covers solutions that, while not part of the exam objectives, have been included with brief coverage as they should be considered required knowledge for a day-to-day Azure role.

NSGs

An **NSG** is a *network security control* and should be part of your DoD approach to protecting the *network layer* from network threats.

An NSG controls access, limits connections to **virtual machines (VMs)** in an Azure **Virtual Network (VNet)**, and uses a *deny-by-default policy*; this means that all access is *denied* unless *explicitly* allowed. The following diagram shows a simplification of this:

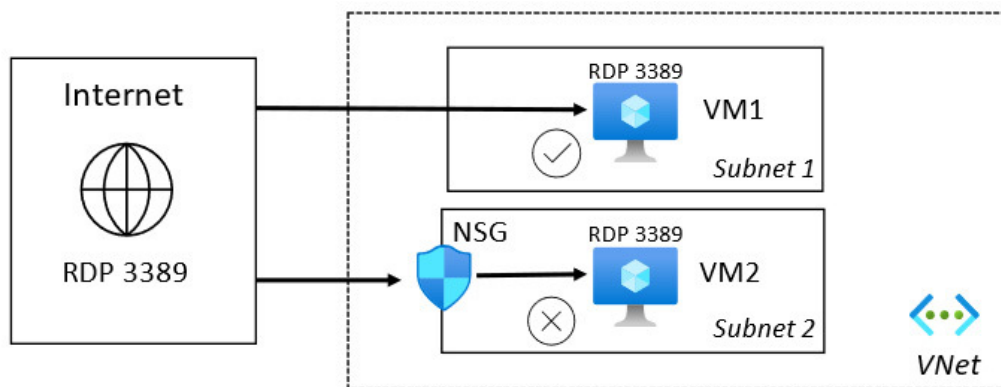


Figure 7.7 – VM access

In the preceding diagram, **Subnet 1** has no traffic filtering in place, so you would be able to connect to Windows **VM1** using **Remote Desktop Protocol (RDP)** on port 3389, and so can an attacker; the most common attack is a brute-force attack to connect to a VM using unsecured management ports—that is, port 3389 for a Windows VM and port 22 for a Linux VM.

Windows **VM2**, however, has an NSG applied at the subnet level and so, by default, will filter traffic and deny access when attempting to connect using **RDP** on port 3389.

An NSG uses a collection of *inbound* and *outbound* rules to *filter network traffic*, in much the same way a traditional packet filter appliance firewall does; it evaluates five data points (*referred to as the 5-tuple method*) to evaluate whether access is *allowed* or *denied*.

An NSG will not encrypt inbound or outbound traffic; it is used for filtering traffic to and from a VM by setting the following five data points:

- *Source* of the traffic
- *Source port* used by the traffic
- *Destination* of the traffic
- *Destination port* used by the traffic
- *Protocol* used by the traffic

The preceding data points will determine if a connection can be made; the NSG will provide an action to be taken by a rule (*allow* or *deny*) and apply a priority. Each rule is given a number—the *lowest-number* rules will be processed first.

Any troubleshooting for not being able to access a VM should start by determining the *ports* and *protocols* required to establish the communication, then identifying if they are being *filtered* or if they are *blocked*.

I use an adage that says: *90% of the time, it's a ports or protocol (or permissions) issue, and so is the other 10%.*

There are a set of default rules for an NSG (which cannot be removed or disabled); these specify which source and destination will be able to access resources to and from the VNets and specify the port and protocol that will be allowed or denied. For a machine to be accessible from the internet on the chosen port, you must ensure that there are rules added to an NSG to allow these ports for communication; by default, all inbound traffic that doesn't meet one of the default rules will be blocked.

An NSG can be associated with a VM **network interface controller (NIC)** and a **subnet** (but not a VNet); the same NSG can be associated with multiple subnets and NICs but a subnet and a NIC can only have one NSG associated. The following diagram aims to visualize this:

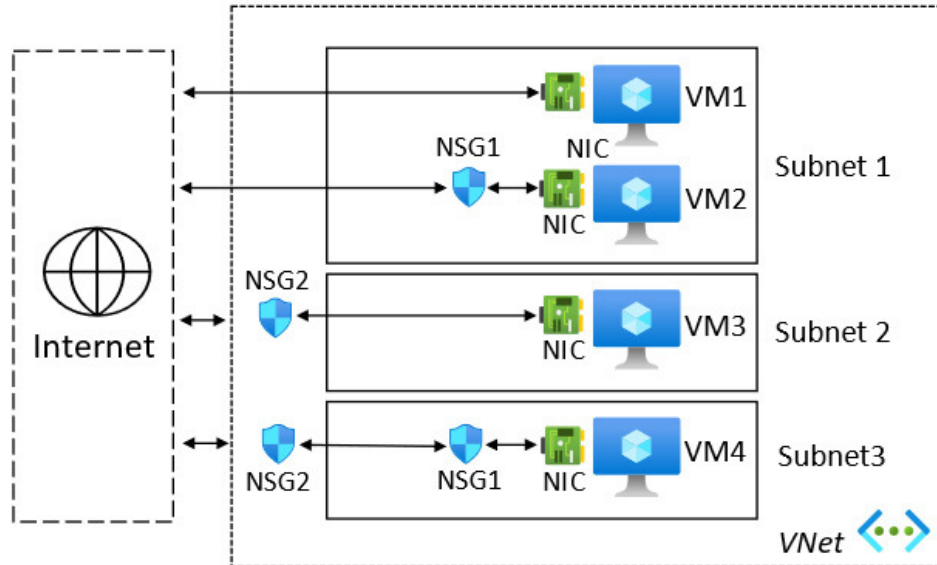


Figure 7.8 – NSG association

An NSG can only be used to control access and filter traffic for resources in the same *region* and *subscription* as the NSG; if you wish to control access and filter traffic for resources across multiple regions or VNets, then Azure Firewall is required for this type of centralized security control of highly distributed networks.

This section looked at NSGs, which can be used as part of a DiD strategy. The following section looks at Azure Firewall, another network security control that can be implemented.

Azure Firewall

Azure Firewall is a cloud-based and Microsoft-managed network security service; it allows centralized (*L3-L7*) connectivity policies and control of network and application traffic across all VNets, across multiple regions and subscriptions. Being an Azure managed service, it has built-in **high availability (HA)**.

It provides control of traffic through **user-defined routing (UDR)** and can create *segmentation of networks* when required for regulatory compliance and when adopting a DiD strategy, as well as implementing a Zero Trust framework. The following diagram provides a typical reference architecture for an Azure firewall to protect resources from attack and control traffic flow:

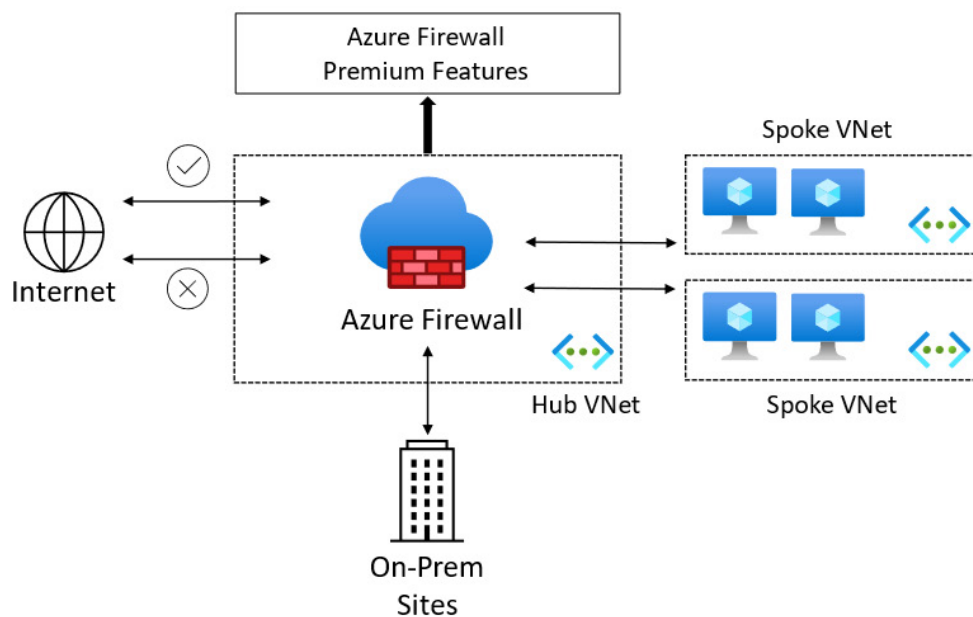


Figure 7.9 – Azure Firewall

The Azure Firewall service is applied at the VNet level and not the VM *network interface* or *subnet level*, as in the case of an NSG. It can filter and control all *incoming* traffic and connections to resources across all VNets that the Azure Firewall service is securing, as well as *outgoing* traffic to other VNets, services, internet, third-party service providers (SPs), and on-prem sites.

Azure Firewall provides inbound **destination network address translation (DNAT)** and outbound **source NAT (SNAT)**; it can have multiple **Public Internet Protocol (IP)** addresses.

Azure Firewall also has a *premium stock-keeping unit (SKU)* that includes next-generation capabilities such as **Transport Layer Security (TLS)** inspection, **Intrusion Detection Systems (IDS)**, **Intrusion Prevention Systems (IPS)**, **Uniform Resource Locator (URL) filtering**, **web categories**; these are requirements of regulated and highly sensitive environments.

In contrast, an NSG can only be applied at the *subnet* or *VM interface* level; the NSG traffic control method can only be associated with a resource in the same region and subscription, so this becomes decentralized and hard to manage and troubleshoot.

It is also important to consider that the Azure Firewall service is not the only method of controlling and securing network and application traffic in Azure; you should also consider **network virtual appliances (NVAs)** from third-party vendors that are available through the Azure Marketplace, such as Barracuda, Fortinet, Palo Alto, WatchGuard, Cisco, SonicWall, and so on.

NVAs are VMs that you create in Azure and are run a vendor's software image of their network appliance to perform a network function such as a **firewall**, **IDS/IPS**, **virtual private network (VPN)**, **software-defined wide-area network (SD-WAN)**, and so on.

Further information and best practices can be found at the following links:

- <https://azure.microsoft.com/solutions/network-appliances>
- <https://azure.microsoft.com/blog/best-practices-to-consider-before-deploying-a-network-virtual-appliance>

This section looked at the Azure Firewall service for securing and controlling network traffic. The following section looks at the Azure DDoS protection service.

Azure DDoS protection

Azure DDoS protection is a cloud-based and Microsoft-managed network security service; it provides protection from network and application attacks that attempt to make a network or application (or any workload) unavailable by flooding it with requests and attempting to exhaust its resources. In addition, it provides *attack analytics* and *attack metrics* reporting.

A single DDoS protection plan is enabled at the *tenant* level and is used across *multiple subscriptions* for cost benefits; by default, the plan is set to *Basic*, which protects resources at no additional cost. This provides protection against common network layer attacks; it requires no application changes or configuration to start making use of this protection.

The DDoS protection plan can also be changed to the *Standard* pricing tier, which provides additional capabilities to protect from *volumetric attacks*, *protocol attacks*, and *resource (application) layer attacks*.

In this section, we looked at the Azure DDoS Protection service. The following sections look at other network and application protection solutions that are available in Azure.

The next section looks at the Azure Key Vault service as a secrets store within Azure.

Azure Key Vault

Azure Key Vault is a cloud-based centralized solution for storing and managing sensitive information used by an *application*, *service*, or *resource* in an encrypted format. It can store information using **hardware security modules (HSMs)** to meet **Federal Information Processing Standard (FIPS) 140-2**.

Azure Key Vault is used for the following:

- **Secrets management:** Used to store information such as *passwords*, *tokens*, **application programming interface (API) keys**, *.pfx files*, and so on
- **Key management:** Used to store cryptographic keys, both software- and hardware-protected.
- **Certificate management:** Used to store and manage *public certification authority (CA) Secure Sockets Layer (SSL)/TLS certificates*

These secrets, keys, or certificates are generally intended to be called programmatically by app or resource.

Similar to all other Azure resources we covered already, Azure Key Vault can be deployed using different approaches, from the Azure CLI/PowerShell, Azure templates, and the Azure portal. Later in this chapter, you will go through an exercise on deploying it from the Azure portal.

Two service tiers are available for Azure Key Vault: *Standard* and *Premium*; the core difference is that only Premium supports HSM-protected keys required to maintain *FIPS 140-2* compliance.

All assets stored in the key vault are encrypted; resources must be in the same region and have the same subscription to store keys in the key vault. From a privacy perspective, Microsoft has no access to the information stored in the key vault or the encryption keys used to encrypt the information held in the key vault.

Access to the key vault is secured through authentication and authorization; security policies can be applied to control access and be monitored. The authentication verifies the identity of the caller of the asset stored in the key vault; this could be a user, a resource such as VM and server apps, a SQL database, an

app service or a function, and so on. The authorization determines which actions they can perform on the requested asset in the key vault, such as *read*, *update*, *delete*, and so on.

This section looked at the Azure Key Vault. The following section looks at securing at the physical resource level through Azure Dedicated Host.

Azure Dedicated Host

With Azure being a *shared resources* computing platform, this means that the default premise is that any VMs you create will run on a *multi-tenant* platform, thus you *share* the underlying physical hardware virtualization hosts with other tenants (organizations). However, your VMs and their workloads are isolated from other workloads on other VMs.

Putting this into an analogy, this means that you may have your own individual hotel room, but you share the hotel building itself with others; you may have your own dedicated front door to access your room, and each person has their own door (as it were) to their own room, but you all share the same corridor, stairs, lifts, restaurant, lobby, front entrance, car park, and so on.

However, Azure Dedicated Host changes that model and provides physical virtualization hosts dedicated to individual customers to host their Azure VMs for Windows and Linux workloads. In our analogy, this means that the building and its contents are yours and that you are not sharing with anybody else. This is not a hotel room—this is a house, a single-tenancy occupied building—and each room (or VM) is yours. The following diagram aims to outline this concept:

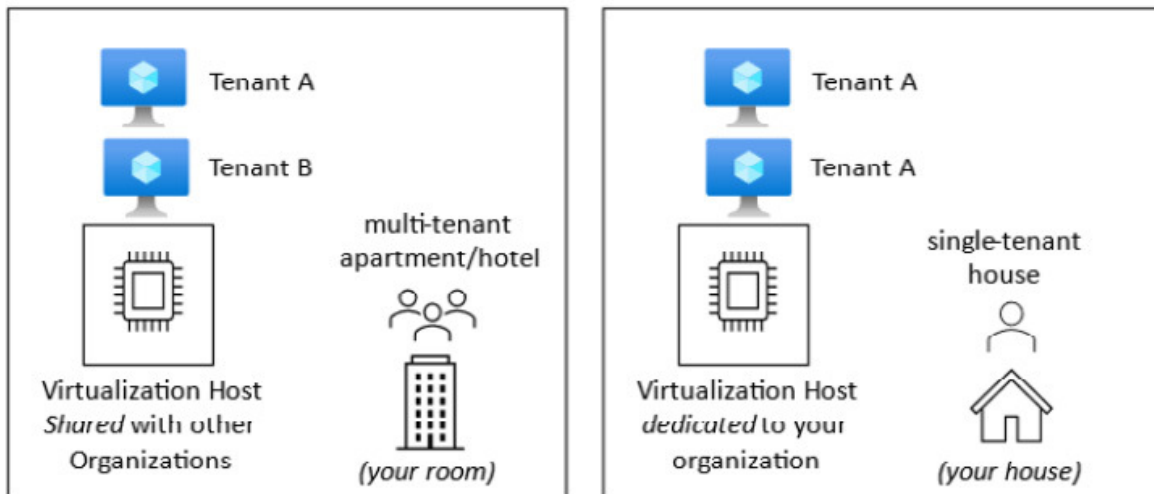


Figure 7.10 – Azure Dedicated Host

Here are some scenarios where you may choose to use Azure Dedicated Host in a solution:

- **Compliance**—An organization may have regulatory compliance that requires only physically dedicated virtualization hosts to run their VMs; they are mandated that they cannot operate their workloads on shared computing resources platforms.
- **Control**—An organization requires control of the compute platform infrastructure that its VMs will run on and visibility into all aspects that they may wish to gain insights on; it also requires control of maintenance windows.
- **Choice**—An organization may wish to have freedom of choice of hardware specifications such as processors, memory, server capabilities, and the VM series and sizes created on the virtualization hosts.

The following aspects are also to be considered:

- **Availability**—This can be provided by a *host group*, allowing multiple dedicated hosts to be provisioned to this group.
- **Pricing**—This is per dedicated host and is not dependent on how many VMs you host on it, but based on the VM family type hosted on it and the region. Storage, networking, and any software are billed separately and not included.
- **Licensing**—This is billed separately and not included.

This section looked at Azure Dedicated Host as a solution for the secure and private virtualization of host hardware. The following section looks at Azure Sentinel, which provides Azure's security operations capabilities.

Azure Sentinel

Azure Sentinel is your *birds-eye view* on centralized security data and events across an organization, using integrated AI for large-scale threat analysis and response.

It is Microsoft's cloud-based **security information and events management (SIEM)** and **security orchestration, automation, and response (SOAR)** tool; it provides security data aggregation, threat analysis, and response across public cloud and on-premises environments.

A SIEM solution collects security log data (*security signaling*) and examines this log data for patterns that could indicate an attack, then correlates event information to identify potentially abnormal activity. Finally, any issues are alerted and this automates responses and remediation. The following diagram illustrates this relationship:

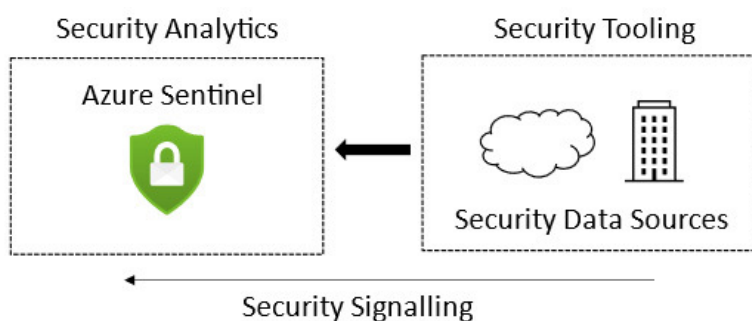


Figure 7.11 – Sentinel positioning

Azure Sentinel provides the following core capabilities:

- *Collects* security data across an organization
- *Detects* threats through AI-powered threat intelligence
- *Investigates* threat-generated critical incidents
- *Responds* through automated reactions and remediations

Azure Sentinel is more than a regular SIEM tool whose core focus is only to provide visibility of threats by collecting security data; the collected data's value is only as

good as the analysis of that data in finding threat and attack patterns.

Azure Sentinel's remit goes beyond that of traditional SIEM solutions. It provides integrated SOAR capabilities that allow you to orchestrate and automate responses once critical incidents occur; all this can occur with unlimited speed and scale that only a public cloud platform such as Azure can provide. The following diagram outlines these capabilities for an **end-to-end (E2E)** security operations solution:

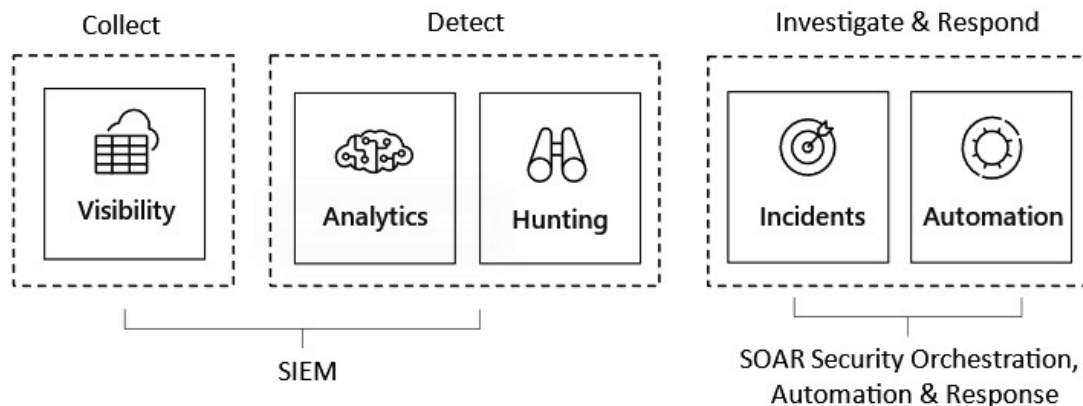


Figure 7.12 – Sentinel security operations proposition

Additional benefits to an organization are that as this is a Microsoft-provided and managed **software-as-a-service (SaaS)** solution, there is no infrastructure setup or maintenance to provide the service. It can unburden teams from non-intelligent, manual SecOps tasks so that they can be retasked on higher-value initiatives and activities.

Implementing Azure Sentinel will increase the *attacker's* attack costs while reducing the *defender's* operations costs of protection from these threats and attacks. Some may question if they can often afford these security solutions; the answer can only be a question itself: *Can your organization afford not to implement these measures?*

Azure Sentinel supports several different ways to collect data, such as connecting to Microsoft solutions natively—for example, **Microsoft 365 (M365)** sources, **Azure Active Directory (Azure AD)**—as well as collecting data from non-Microsoft security data sources, along with any source at another cloud provider, SP, or on-

prem that uses **System Logging Protocol (Syslog)**, **Common Event Format (CEF)**, or that has a **REpresentational State Transfer (REST) API**.

After connecting the security data sources to Azure Sentinel, the **Azure Log Analytics** service is used; the *Log Analytics workspaces* act as the data store for collecting and retaining logs.

In this section, we looked at security operations using Azure Sentinel. The following section looks at the Azure Security Center service, which provides Azure's security-posture management capabilities.

Azure Security Center

Azure Security Center is Microsoft's **Cloud Security Posture Management (CSPM)** tool; it provides *security policy and compliance management, actionable security hardening tasks, and secure scores*.

A security posture is an organization's threat protection and response capabilities; this ensures that an organization has the ability for its systems, data, and identities to be recoverable and operational should an attack be successful. Adopting the Azure Security Center best practices and recommendations can help increase an organization's security posture and secure score.

A security posture's goal should be to reduce the exposure to threats, shrinking attack-surface areas and vectors while building resilience to attacks, gaining intelligence, and learning from each attack, as they cannot be prevented or eliminated. It is critical to understand that an attacker only has to be successful once, while you must protect everything, all the time.

Azure Security Center has more value to an organization than just providing recommendations and best practices; it is hybrid by design and supports an organization adopting a Zero Trust strategy and a DoD approach to protecting resources.

This is all provided from a single place for actionable insights and reports of non-conformity against any required security controls, policies, or mandated regulatory compliance standards.

The **Secure Score** feature in Azure Security Center measures an organization's security posture; it is based on security recommendations and controls, whereby your score is determined by the number of security recommendations and controls you meet.

The secure score will provide you information on your current score, the maximum score available, and the potential increase by implementing the recommendations provided, as well as finding gaps for improving your score and comparing with industry standards and regulatory compliance that you may wish to conform to,

such as the **International Organization for Standardization (ISO) 27001** or the **Payment Card Industry (PCI)**; these reports can also be exported.

The *continuous assessments*, *security recommendations*, and *secure score* are part of the Free-tier capabilities enabled by default within Azure Security Center.

Azure Security Center can also be integrated with **Azure Sentinel** and **Azure Defender**; this allows an organization to follow a layered approach to security using independent tools, seamless integration, and the ability to share threat intelligence and resources' signal data. You will require Azure Defender to be enabled, which has cost implications outside of the Azure Security Center service. The following diagram aims to visualize this approach:

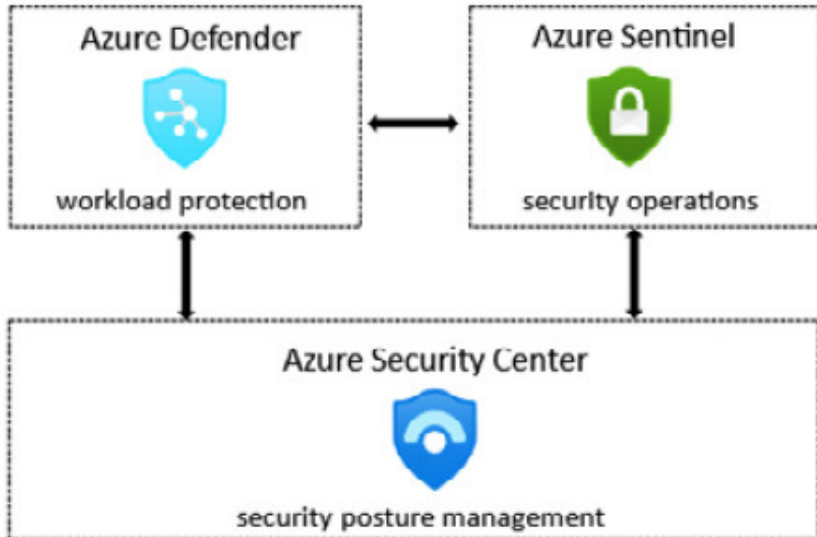


Figure 7.13 – Security center positioning

Azure Security Center can be considered the *foundational* or *base layer* in this integrated security framework, which can then feed into and enrich other security services such as Azure Sentinel and be fed and enriched by integrations from other security services such as Azure Defender, which provides a security information continuum.

Azure Security Center can operate as a Free pricing tier option; this is also known as *Azure Defender Off mode*. Additional capabilities are also available as part of Azure Defender integration but are not included in the free-of-charge tier; this is

also known as *Azure Defender On mode*. As mentioned, this will have cost implications.

Only available in *Azure Defender-enabled mode* are security alerts generated for resources; these are triggered through advanced threat detections on your resources. Rather than view individual alerts, Azure Security Center uses a smart-alert correlation of the different alerts and creates security incidents, a collection of related alerts; this gives a single unified view of an attack chain.

The Azure Security Center dashboard, as well as displaying alerts and policy and compliance status, also has a section named **Resource security hygiene**; this provides a dashboard view of the security recommendations for all resources, showing those resources with the highest occurrence of recommendations and the highest-impact recommendations.

This section looked at adopting Azure Security Center to improve an organization's security posture. The following section looks at some further solutions that, while not part of the exam objectives, have been included with brief coverage as they should be considered required knowledge for a day-to-day Azure role.

Other protection solutions

Azure, in addition, provides the following two services that provide critical application-delivery protection and security:

- **Azure Application Gateway** provides a secure way of load balancing to service endpoints such as VMs and can be used for SSL termination, also called SSL offloading. Azure Application Gateway also provides **web application firewall (WAF)** capabilities to protect internet-facing web applications from application-layer attacks (L7); distribution of traffic is to endpoints within a *region only*.
- The **Azure Front Door** service provides an **application delivery network (ADN)** service. It also provides layer 7 (application layer) load balancing, WAF, and content acceleration capabilities for applications; however, it operates at the *geographic layer* and is *not regional*.

While both Front Door and Application Gateway are layer 7 (*HTTP/HTTPS*) load balancers, the primary difference is that Front Door is a *global app delivery service*; *in contrast*, Application Gateway is a *regional app delivery service*.

In summary, *HTTP/HTTPS (layer 7)* secure load balancing and content delivery is performed by both services; the key differentiator is the layer at which they operate—*regional or global*.

You can understand more about these two services and when it is appropriate to use each in a solution at the following link:

<https://docs.microsoft.com/azure/architecture/guide/technology-choices/load-balancing-overview>.

It is also important to note that these solutions are not part of the exam objectives but were considered critical to be included.

In this section, we looked at other network and application protection solutions that are available in Azure but not part of the exam objectives. The following section looks at hands-on exercises to further build on your skills learned in this chapter.

Hands-on exercise

To support your learning with some practical skills, we will look at the hands-on creation of some of the resources covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – Create an Azure key vault.
- Exercise 2 – Secure network access using an NSG.

Getting started

To get started with these hands-on exercises, you will need an Azure subscription that has access to create and delete resources in the subscription. You can use an existing account that you created as part of the exercises from any chapter in this book; alternatively, you can create a free Azure account from this link:

<https://azure.microsoft.com/free>.

This free Azure account provides the following:

- 12 months of free services.
- USD \$200 credit to explore Azure for 30 days.
- 25+ services that are always free.

Exercise 1 – Create an Azure key vault

This section will look at installing Azure Key Vault and then storing a secret, which will be a password in this exercise.

In the following sub-sections, you can see the procedure to complete the exercise, segregated into tasks for a better understanding.

Task: Access the Azure portal

1. Log in to the Azure portal at <https://portal.azure.com>. You can alternatively use the Azure desktop app, found at <https://portal.azure.com/App/Download>.

Task: Create an Azure key vault

2. In the search bar, type in **key vaults**; click **key vaults** from the results list.
3. From the **Key vaults** blade, click the **+ Create** button on the top toolbar.
4. From the **Basics** tab, set project details settings as required for the subscription and the resource group.
5. Set **Instance details** settings, as follows:
 - **Key vault name**—Enter a name.
 - **Region**—Set as required.
 - **Pricing tier**—Leave set to **Standard**.
6. Set **Recovery options** settings, as follows:
 - **Days to retain deleted vaults**—Leave set to 90 days.
 - **Purge protection**—Leave set to **Disabled**.
7. Click **Next: Access policy**; leave at default settings.
8. Click **Next: Networking**; leave at default settings.
9. Click **Review + create**.
10. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.

11. When the deployment is complete, you will receive a notification that the deployment succeeded.
12. Click **Go to resource** from the **Deployment** blade; alternatively, navigate to the Azure Key Vault instance.

Task: Add a secret to the key vault

13. From the **Key Vault** blade of the instance created, click **Secrets** under **Settings**.
14. Click **+ Generate/Import**.
15. From the **Create a secret** blade, set the following:
 - **Upload options**—Leave set to **Manual**.
 - **Name**—Enter a name for this secret.
 - **Value**—Enter the secret you wish to store (that is, a password).
 - **Content type**—Set as required.
 - **Set activation date**—Set as required.
 - **Set expiration date**—Set as required.
 - **Enabled**—Leave as **Yes**.

16. Click **Create**.

17. You will receive a notification that the secret was successfully created.

18. You will now see the secret listed in secrets under **Settings** for the key vault instance.

In this exercise, we successfully created an Azure key vault and created a secret. In the following exercise, we will look at securing network access using an NSG.

Exercise 2 – Secure network access using an NSG

This section will look at creating an NSG and associating with a subnet; adding an inbound rule, allowing us to connect to a VM over RDP while restricting connections to access from a known IP address only; then, creating an outbound rule to deny internet access.

For this exercise, we will create a new VM.

In the following sub-sections, you can see the procedure to complete the exercise, segregated into tasks for a better understanding.

Task: Access the Azure portal

1. Log in to the Azure portal at <https://portal.azure.com>. You can alternatively use the Azure desktop app, found at <https://portal.azure.com/App/Download>.

Task: Create a VM

2. In the search bar, type in **virtual machines**; click **Virtual machines** from the list of services.
3. From the **Virtual machines** blade, click **+ Create** and then **Virtual machine** from the top menu of the blade.
4. Set **Project and Instance details** and **Administrative accounts** as required.
5. Set **Public inbound port rules** to **None**.
6. Set **Licensing** at the default of unchecked (that is, **No**) to use an existing Windows Server license.
7. Click **Next: Disks**.
8. Leave set at defaults.
9. Click **Next: Networking**.
10. From **Network interface**, set the following:
 - **Virtual network**—Use the default provided.
 - **Subnet**—Use the default provided.
 - **Public IP**—Click **Create new**, then enter a name and click **OK**.
 - **NIC network security group**—Select **None**.
 - Leave all other settings at defaults.

11. Click **Next: Management**.
12. Leave set at defaults.
13. Click **Next: Advanced**.
14. Leave set at defaults.
15. Click **Next: Tags** and add any tags as required; click **Next: Review + create**.
16. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.
17. You will receive a notification that the resource was created successfully.

Task: Create an NSG

18. In the search bar, type in **network security groups**; click **network security groups** from the list of services.
19. From the **Virtual machines** blade, click **+ Create** from the top menu of the blade.
20. Set **Project and Instance details** and **Administrative accounts** as required.
21. Click **Next: Tags** and add any tags as required; click **Next: Review + create**.
22. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.
23. You will receive a notification that the resource was created successfully.
24. Click **Go to resource** from the deployment blade; alternatively, navigate to the Azure NSG instance.

Task: Associate NSG to a subnet

25. From the **Created NSGs** blade of the instance created, click **Subnets** under **Settings**.
26. Click **+ Associate** from the top toolbar.
27. Select the VNet and subnet of the VM you created in the previous exercise from the **Associate subnet** blade.
28. Click **OK**.
29. You will receive a notification that the changes were saved successfully.

Task: Attempt to connect to a VM using RDP

30. Navigate to the VM created in this exercise.

31. From the **Overview** pane, click **Connect** and then click **RDP**.
32. From the **Connect** pane, click **Download RDP file**.
33. Open the downloaded file and click **Connect**.
34. You will see a message box that says **Remote Desktop can't connect to the remote computer**. This is to be expected, and we will resolve this in the next task to allow a connection.

Task: Add an inbound rule to allow RDP access

35. From the **Virtual machines** blade, click **Networking** under **Settings**. You will see that from the inbound port rules, all inbound connections are denied unless their source is the VNet or Azure Load Balancer.
36. From **Inbound port rules** blade, click **Add inbound port rule**.
37. Open a browser and from Google, type **what's my ip** and note your IP for the next step.
38. From the **Add inbound security rule** blade, leave all other options at their defaults apart from the following:

- **Source**—Select **IP addresses**.
- **Source IP addresses/CIDR ranges**—Set to your IP from the previous step.
- **Service**—Select **RDP**.
- **Action**—Ensure **Allow** is set.
- **Name**—Provide a name, such as **AllowInbound_RDP_KnownIP**.
- **Description**—Enter as required.

39. Click **Add**.
40. You will receive a notification that the rule was created successfully.
41. Open the downloaded file again and click **Connect**.
42. This time, you will be prompted to enter your credentials and will be able to connect to the VM successfully.

Task: Access the internet from the VM

43. While logged in to the VM, open a browser and confirm you have reached the internet by visiting a site such as www.milesbetter.solutions (you may need to adjust **Internet Explorer (IE)** security settings). We will restrict this access to the internet in the next exercise.

Task: Add an outbound rule to deny internet access

44. From the **Virtual machines** blade, click **Networking** under **Settings**; you will see from the **Outbound port rules** tab that all outbound connections are allowed to the internet.

45. From **Outbound port rules** blade, click **Add outbound port rule**.

46. From the **Add outbound security rule** blade, leave all other options at default apart from the following:

- **Destination**—Select **Service tag**.
- **Destination service tag**—Select **Internet**.
- **Destination port ranges**—Enter the symbol * (asterisk symbol).
- **Action**—Ensure **Deny** is set.
- **Priority**—Enter a value of 200 .
- **Name**—Provide a name, such as **DenyOutbound_Internet** .
- **Description**—Enter as required.

47. Click **Add**.

48. You will receive a notification that the rule was created successfully.

49. From the VM, open a browser again and confirm you can no longer reach the internet by visiting a site such as www.milesbetter.solutions.

50. This time, you will see a message from the browser such as **Can't reach this page**.

In this exercise, we successfully created an NSG and associated with a subnet; we added an inbound rule that allows us to connect to a VM over RDP while restricting connections to be able to access from a known IP address only; then, we created an outbound rule to deny internet access.

This section covered hands-on exercises. The following section provides a summary of this chapter.

Summary

This chapter, *Azure Security*, included coverage of the *AZ-900 Azure Fundamentals* exam skills area: *Describe general security and network security features*.

In this chapter, you have learned about security concepts, security services, and security operations. This included skills that will provide you the confidence to explain and discuss the functionality and usage of the following aspects with a business or technical audience: *the concepts of threat modeling, DoD, Azure Key Vault, Azure Dedicated Host, NSGs, Azure Firewall, Azure DDoS Protection, Azure Security Center, and Azure Sentinel*.

The following chapter will cover identity services, including **Azure AD, Active Directory (AD)**, hybrid identity authentication, authorization, **single sign-on (SSO)**, **multi-factor authentication (MFA)**, and Conditional Access.

Additional information and study references

This section provides links to additional exam information and study references, as follows:

- *Exam AZ-900: Microsoft Azure Fundamentals:*

<https://docs.microsoft.com/learn/certifications/exams/az-900>

- *Exam AZ-900: Microsoft Azure Fundamentals – Skills Measured:*

<https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>

- *Microsoft Learn: Azure Fundamentals part 4 – Describe general security and network security features:*

<https://docs.microsoft.com/learn/paths/az-900-describe-general-security-network-security-features>

Skills check

Challenge yourself with what you have learned in this chapter by answering the following questions:

1. Explain the Zero Trust framework principles and foundational elements.
2. Explain a DoD strategy; what are the goals?
3. What are the three uses for Azure Key Vault?
4. When would Azure Dedicated Host be required for a solution?
5. Which Azure security feature can be used to improve security posture?
6. Which Azure security feature can be used for Azure and on-premises security operations?
7. Are all Azure security capabilities included in the Free pricing tier?
8. How is Azure Sentinel positioned against Azure Security Center and Azure Defender?
9. How are NSGs positioned against Azure Firewall?
10. When would the Azure DDoS protection service be used in a solution?

Section 5: Identity, Governance, Privacy, and Compliance

This section provides complete coverage of the knowledge and skills required for the *skills measured* of the *Describe identity, governance, privacy, and compliance features* section of the exam. We also cover knowledge and skills that go beyond the exam content, so you are prepared for a real-world, day-to-day Azure-focused role.

This part of the book comprises the following chapters:

- [Chapter 8](#), *Azure Identity Services*
- [Chapter 9](#), *Azure Governance*
- [Chapter 10](#), *Azure Privacy and Compliance*

Chapter 8: Azure Identity Services

In [Chapter 7](#), *Azure Security*, you learned skills that covered the security aspects of Azure, including security concepts, the security services themselves that you can enable, and security posture management and security operations tooling.

This chapter will outline the identity services in Azure, including **Azure Active Directory (Azure AD)**, **Active Directory (AD)**, **hybrid identity authentication, authorization, Single Sign-On (SSO)**, **Multi-Factor Authentication (MFA)**, and **Conditional Access**.

This chapter aims to provide coverage of the AZ-900 Azure Fundamentals Skills Measured section: *describe identity, governance, privacy, and compliance features*.

By the end of this chapter, you will have learned how to do the following:

- Define Azure AD.
- Describe the functionality and usage of Azure AD.
- Explain the difference between authentication and authorization.
- Describe the functionality and usage of Conditional Access, MFA, and SSO.

To support your learning with some practical skills, we will also look at the hands-on usage of some of the tools covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – creating a new tenant instance of Azure AD
- Exercise 2 – creating users and groups in Azure AD

In addition, this chapter's goal is also to take your knowledge beyond the exam objectives, so you are prepared for a real-world, day-to-day Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will require the following:

- An Azure subscription that has access to create and delete resources in the subscription. If you do not have an Azure subscription, you can create a free Azure account from this URL: <https://azure.microsoft.com/free>.
- Access to an internet browser. You will need to log in to the Azure portal: <https://portal.azure.com>.
- You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.

Azure AD

Azure AD can be thought of primarily as a cloud-based centralized **Identity Provider (IDP)** and *directory service* for *objects*.

Azure AD is the foundation of granting access to resources through **Identity and Access Management (IAM)** for cloud and hybrid environments and providing authentication and authorization for users, apps, and devices.

Objects are stored in Azure AD with attributes; for user identities, the core attributes would be their sign-in name, known as their **User Principal Name (UPN)**, password, location, assigned roles, group membership, devices, licenses, and authentication methods. The following diagram aims to visualize *Azure AD* as the *centralized cloud IDP*:

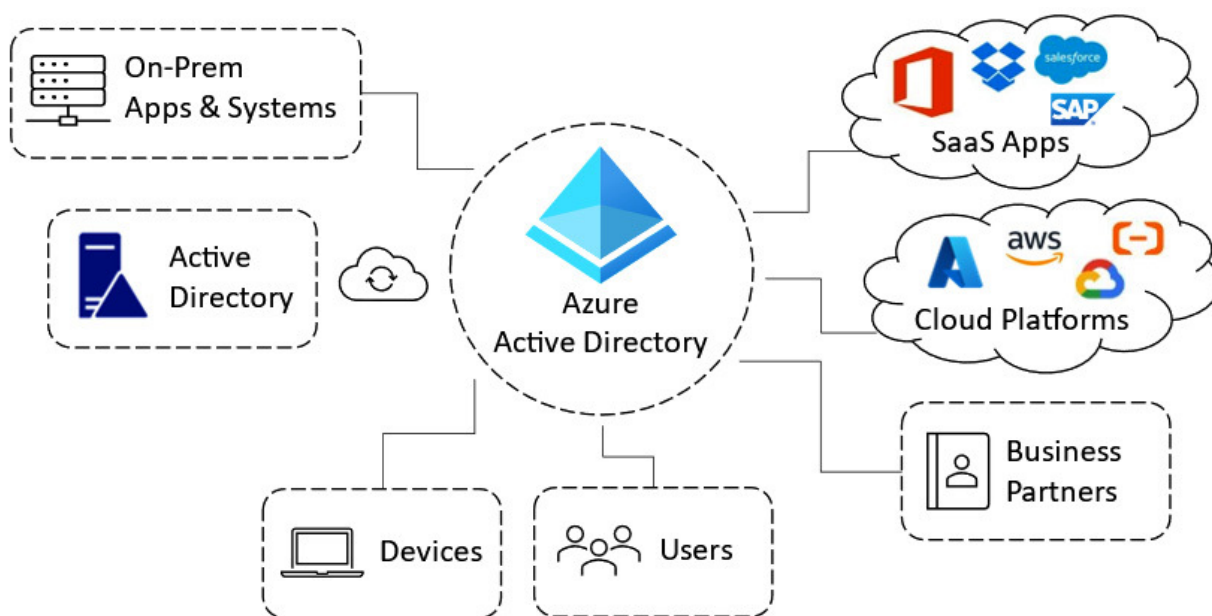


Figure 8.1 – Active AD

In addition to an organization's user management, Azure AD provides the following:

- Device management
- Application management
- **Business to Business (B2B)** and **Business to Customer (B2C)** identity services

- SSO
- MFA and Conditional Access

An instance of Azure AD (*referred to as a directory*) is created for each Microsoft tenant that is created; that is, each `<tenant>.onmicrosoft.com` represents an instance of Azure AD.

You will only have one directory per `<tenant>.onmicrosoft.com`, and by creating a new `<tenant>.onmicrosoft.com`, you will be creating a new directory.

There are four editions of Azure AD. They are as follows:

- **Free:** This edition is included when you create a new tenant and is created with the provisioning of a Microsoft online service such as Microsoft 365, Dynamics 365, and Azure.
- **Office 365 Apps:** This edition is included with Microsoft 365. It includes a **Service-Level Agreement (SLA)** of 99.9% availability and additional functionality such as organization branding and two-way synchronization of objects between AD and Azure AD.
- **Premium P1 and P2:** These editions provide additional identity protection and identity governance functionality on top of the basic functionality included in the free and Microsoft 365 editions.

Azure AD has objects that are referred to as *security principals* that form the basis for identities. They can be one of the following types:

- **User:** An entity that Azure AD can manage; this user can be a member of the organization's tenancy or a *guest user* that does not belong to your organization.

Azure AD supports *guest users* through a feature called *B2B*. This allows access to resources in your organization's tenancy for users that are not part of your organization, such as business partners. Azure AD also supports *B2C*, allowing access to Azure AD resources via an *external IDP account* such as from *Facebook* or *Google*.

- **Application service principal:** An entity that represents an identity of a service or application in Azure.
- **Managed identity service principal:** An entity representing a special kind of service principal identity for a service or application to use in place of a user identity; there are *system-assigned* and *user-assigned* managed identities.
- **Device:** A physical entity: laptop, tablet, phone, virtual machine, and so on.

When is AD not AD? When it's Azure AD.

Both AD and **Azure AD** are IDPs, and while they share *AD* in the name, they function very differently.

AD was introduced in 2000 as Microsoft's directory service. It is a role that is installed as part of Windows Server, and servers running this role are referred to as **Domain Controllers (DCs)**. It allows access to multiple resources that are stored as *computer objects* within the directory service, with identities stored as *user objects*.

AD is not a single function; it has several services that can be provided, the core services being *AD Domain Services*, *AD Certificate Services*, and **AD Federation Services (ADFS)**. Many of these services are still required and will still be around for some years to come, and it should be noted that Azure AD cannot be considered a 100% replacement for AD; although it can modernize your IAM strategy and approach so these *legacy services* could be replaced and retired through phasing out, as you move to modern identity services. The following diagram aims to visualize this:

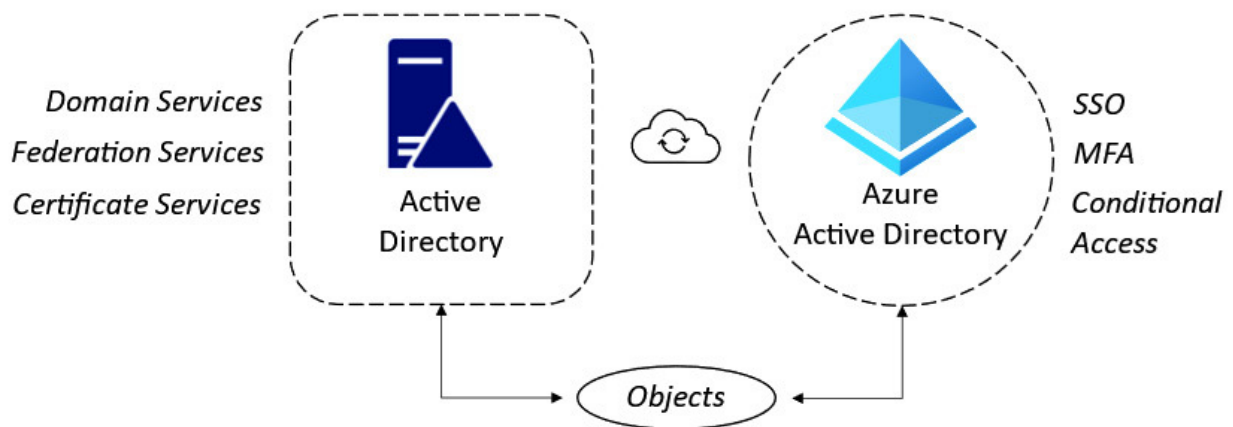


Figure 8.2 – AD and Azure AD

There is a misconception that Azure AD is the cloud equivalent of the traditional Windows Server-based AD; this is not the case, and it is important to understand that Azure AD is not cloud DCs and cannot replace the functionality provided by traditional implementations of Windows Server AD.

Unlike AD, there is nothing to install for Azure AD, and no DCs are required while still allowing devices to appear as objects in the directory; Microsoft provides Azure AD as a fully managed IDP platform provided as **Software-as-a-Service (SaaS)**.

AD can be connected to Azure AD using Azure AD Connect, a free download tool to allow an organization to establish a *hybrid identity*. This tool synchronizes user identities, attributes, and objects between both *IDPs*.

A hybrid identity approach allows users to access resources in Azure AD using their AD identity; the same username and password are used to access resources accessed in both IDP environments.

More information on hybrid identities and Azure AD Connect can be found at this URL: <https://docs.microsoft.com/azure/active-directory/hybrid/whatis-azure-ad-connect>.

In this section, we looked at Azure AD. The following section looks at authentication and authorization.

Authentication and authorization

Accessing resources is based on a two-stage concept that consists of firstly *authenticating* and then *authorizing*; in a nutshell, *identifying who you are* and *determining what you can do*.

Authentication, also referred to as **AuthN**, is the process of establishing the identity of a person (*or service*) and proving they are who they say they are. This can be done by validating provided access credentials information against stored or known identifying information.

Authorization, also referred to as **AuthZ**, is the process of establishing what level of access the authenticated person (*or service*) has to the resource; that is, what they can access and what actions they may perform:

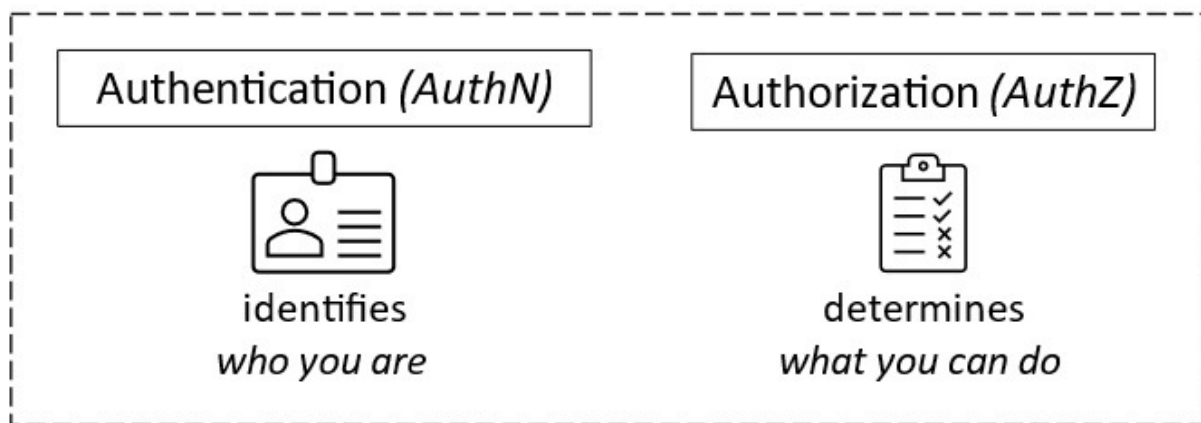


Figure 8.3 – Authentication and authorization

In this section, we looked at the concepts of authentication and authorization. The following section looks at SSO.

Single sign-on

SSO means only needing one set of credentials that you enter once to access all resources enabled to use SSO in your organization; you are not prompted to sign in again.

In addition, user provisioning to apps is accelerated with just-in-time access for new hires and temporary staff and allows a governed leavers process when users no longer need access to an app.

You configure Azure AD as the trusted IDP for each app you wish to enable SSO through a centralized portal. These apps can be cloud apps, public cloud provider platforms, as well as on-premises apps. You can enforce secure access with identity protection through MFA and Conditional Access, as well as risk-based access policies.

In this section, we looked at SSO. The following section looks at MFA and Conditional Access.

MFA and Conditional Access

MFA (which includes **Two-Factor Authentication (2FA)**) provides an additional layer of security for identifying a user by requiring the user to submit two or more elements for authentication. MFA is based on the following principles:

- **Knowledge:** Something that only the user *knows*, such as a *password* or *pin*.
- **Possession:** Something that only the user *has*, such as a *code sent to a phone*, a *token*, or a *key*.
- **Inherent:** Something that only the user *is*, such as *biometrics*.

Conditional Access works alongside MFA to provide more granular levels of access control; information is collected from the *sign-in process (signals)*, and then decisions are made upon that information to determine whether access to the requested resource will be granted or denied and whether the user will require additional factors of authentication or require taking other action, such as resetting their password. This is visualized in the following diagram:

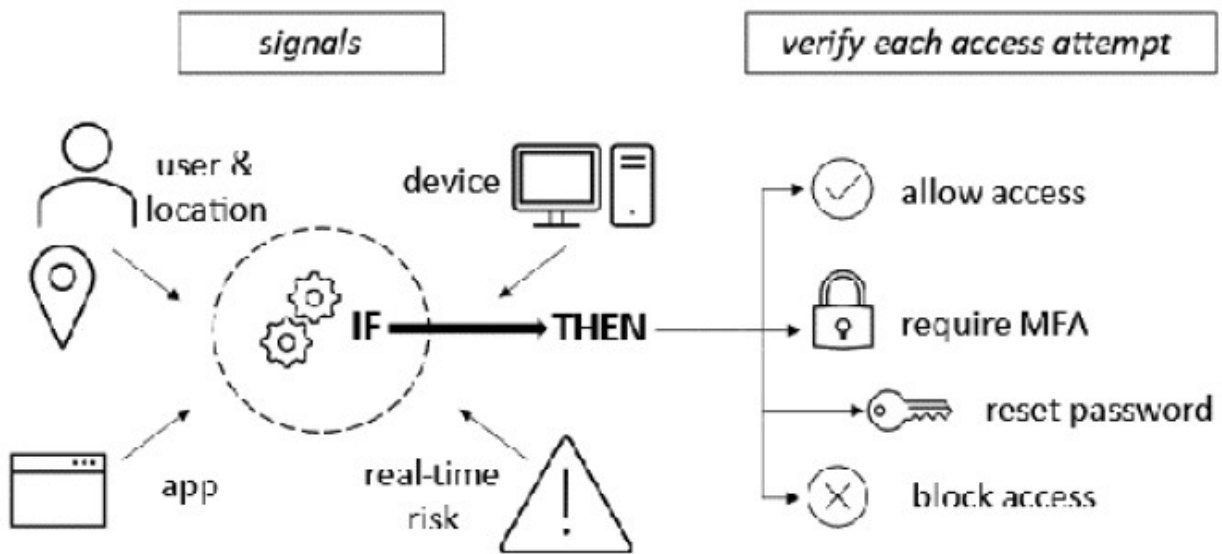


Figure 8.4 – Conditional Access

The user signal information could be the app trying to get access, the device or location being accessed from, or the security updates status. This could mean that *if* the user is trying to access the resource from a location that is unknown, a non-

managed device/personal device, or a device that doesn't have the latest security updates, *then* they will be required to authenticate with a second factor of authentication or even be denied access to the resource.

Conditional Access is a licensed function that requires an *Azure Premium P1 or P2 license*; this functionality is also included in the *Microsoft 365 Business Premium license*.

In this section, we looked at MFA and Conditional Access. The following section looks at hands-on exercises to further build on your skills learned in this chapter.

Hands-on exercises

To support your learning with some practical skills, we will look at hands-on examples of some of the topics covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – creating a new tenant instance of Azure AD
- Exercise 2 – creating users and groups in Azure AD

Getting started

To get started with the hands-on exercises, you will need an Azure subscription that has access to create and delete resources. You can use an existing account that you created as part of the exercises from any chapter in this book. Alternatively, you can create a free Azure account from this URL: <https://azure.microsoft.com/free>.

This free Azure account provides the following:

- 12 months of free services
- \$200 credit to explore Azure for 30 days
- 25+ services that are always free

Exercise 1 – creating a new tenant instance of Azure AD

This section will look at creating a new tenant instance of Azure AD using the Azure portal.

In the following sub-sections, you can see the procedure to complete the exercise, segregated into tasks for better understanding.

Task – accessing the Azure portal

1. Log in to the Azure portal: <https://portal.azure.com>. You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – creating a new tenant for an organization

2. In the search bar, type in **Azure Active Directory**. Click **Azure Active Directory** from the results list.
3. In the Azure AD **Overview** blade, click **Manage tenants** on the top toolbar.
4. In the **Switch tenant** blade, click **+ Create** on the top toolbar.
5. In the **Create a tenant** blade, on the **Basics** tab, leave **Tenant type** set to **Azure Active Directory**.
6. In the **Configuration** tab, set the following as required:
 - **Organization name**
 - **Initial domain name**
 - **Country/Region**
7. Click **Next: Review + create**.
8. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.
9. You will receive a notification that the task succeeded, and the new tenant and an instance of Azure AD were created.
10. Your account performing this task becomes the first user of this tenant and will automatically be assigned the *Global Admin* role.

In this exercise, we successfully created a new tenant and an instance of Azure AD.
In the following exercise, we will look at creating users and groups on Azure AD.

Exercise 2 – creating users and groups in Azure AD

This section will look at creating users and groups for Azure AD using the Azure portal. You can use an existing Azure AD instance to perform this exercise if you wish.

In the following sub-sections, you can see the procedure to complete the exercise, segregated into tasks for a better understanding:

Task – accessing the Azure portal

1. Log in to the Azure portal: <https://portal.azure.com>. You can alternatively use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – creating a new user

2. In the search bar, type in **users**. Click **Users** from the results list.
3. In the **Users** blade, click **+ New user** on the top toolbar.
4. On the **New user** page, ensure the **Create user** option is selected.
5. Set the **Identity** settings as follows:
 - **User name:** *Enter a name*. This will be the username for the user to sign in with (this will form their UPN). If you have added custom domain names to Azure AD, these will appear here in the dropdown; select the domain name you wish to give the user to sign in with.
 - **Name:** *Set as required*. This is a descriptive name for the user.
 - **First name** and **Last name:** *Optional*.
6. For the **Password** settings, select as required.
7. For the **Groups and Roles** setting, select as required.
8. For the **Block sign-in** setting, leave it at the default setting.
9. For the **Usage location** setting, set the location of the user as required.
10. For the **Job Info** setting, enter as required or leave at the default setting.
11. Click **Create**.

You will receive a notification that the user was successfully created.

12. You will now see the user listed in the **Users** blade; the **User principal name** column is what the user will enter to sign in.

Task – creating a new guest (B2B) user

13. In the search bar, type in **users**. Click **Users** from the results list.

14. In the **Users** blade, click **+ New guest user** on the top toolbar.

15. On the **New user** page, ensure the **Invite user** option is selected.

16. Set the **Identity** settings as follows:

- **Name:** *Set as required.* This is a descriptive name for the user.
- **Email address:** *Set as required.* This will be where the invite is sent and must be associated with a Microsoft account to allow access.
- **First name** and **Last name:** *Optional.*

17. For the **Personal message** setting, select as required.

18. For the **Groups and Roles** setting, select as required.

19. For the **Block sign-in** setting, leave it at the default setting.

20. For the **Usage location** setting, set the location of the user as required.

21. For the **Job Info** setting, enter as required or leave at the default setting.

22. Click **Invite**.

You will receive a notification that the user was successfully invited.

23. You will now see the user listed in the **Users** blade; the **User principal name** column is what the user will enter to sign in.

24. The invited guest user will now receive an email to accept the invitation.

25. The invited guest user will sign in with the Microsoft AD tenant associated with the email address.

Task – creating a group and adding a user

26. In the search bar, type in **groups**. Click **Groups** from the results list.

27. In the **Groups** blade, click **+ New Group** on the top toolbar.

28. Select the group type as required. Leave as the default of **Security** for this exercise.

29. Enter a group name as required.

30. Enter a group description as required.

31. Set an Azure AD roles assignment as required. Leave as the default of **No** for this exercise.
32. Select the membership type as required. Leave as the default of **Assigned** for this exercise.
33. Set **Owners** as required. Leave as the default of **No owners selected** for this exercise.
34. Set **Members** as required. Select the created new user from the previous task for this exercise.
35. Click **Create**.

You will receive a notification that the group was successfully created.

36. You will now see the group listed in the **Groups** blade.
37. Click the group you created. From the **Overview** screen, you will see the number of users in this group.
38. Click **Members** from the **Manage** section on the left navigation toolbar of the group screen.
39. From the **Members** page for the group, you will see the users added to this group. You can remove users from the top toolbar and perform bulk operations.

In this exercise, we successfully created a new user and a new guest user in Azure AD, and then created a new group and added a user.

This section covered hands-on exercises. The following section provides a summary of this chapter.

Summary

This chapter on Azure identity services included coverage of some of the topics for the AZ-900 Azure Fundamentals exam skills area: describe identity, governance, privacy, and compliance features.

In this chapter, you have learned the definition of Azure AD and described its functionality and usage, the difference between authentication and authorization, and the functionality and usage of Conditional Access, MFA, and SSO.

Further knowledge beyond the exam objectives was provided to prepare for a real-world, day-to-day Azure-focused role.

Assigning **Role-Based Access Control (RBAC)** will be covered in the following chapter.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure fundamentals

<https://docs.microsoft.com/learn/certifications/exams/az-900>

- Exam AZ-900: Skills outline

<https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>

- Microsoft Learn: Azure Fundamentals – Describe identity, governance, privacy, and compliance features

<https://docs.microsoft.com/learn/paths/az-900-describe-identity-governance-privacy-compliance-features/>

Skills check

Challenge yourself with what you have learned in this chapter:

1. Explain Azure AD.
2. List four features provided by Azure AD.
3. Explain the difference between the different Azure AD editions.
4. Explain the difference between the four security principals.
5. Explain how Azure AD differs from AD.
6. List three services that AD provides that Azure AD does not.
7. Provide a scenario where you need to synchronize AD with Azure AD.
8. Explain what is meant by AuthN and AuthZ.
9. Explain what is meant by SSO.
10. Explain the principles MFA is based on.

Chapter 9: Azure Governance

In [Chapter 8](#), *Azure Identity Services*, you learned the skills that covered the identity and access management aspects available in Azure, including **Azure Active Directory (AD)**, **Active Directory**, **hybrid identity authentication**, **authorization**, **single sign-on**, **multi-factor authentication**, and **Conditional Access**.

This chapter will outline various governance services in Azure, including **resource tags**, **resource locks**, **role-based access control (RBAC)**, **Azure Policy**, **Azure Blueprints**, and the **Cloud Adoption Framework (CAF) for Azure**.

This chapter aims to provide coverage of the AZ-900 Azure Fundamentals Skills Measured section known as *Describe identity, governance, privacy, and compliance features*.

By the end of this chapter, you will be able to do the following:

- Describe the functionality and usage of resource tags, resource locks, RBAC, Azure Policy, and Azure Blueprints.
- Describe the CAF for Azure.

To support your learning with some practical skills, we will also look at the hands-on usage of some of the tools covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – assigning access with RBAC
- Exercise 2 – creating a custom RBAC role
- Exercise 3 – adding a resource lock to a resource group
- Exercise 4 – enabling resource tagging with Azure Policy
- Exercise 5 – limiting the resource creation location with Azure Policy

In addition, this chapter's goal is to take your knowledge beyond the exam objectives so that you are prepared for a real-world, day-to-day Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will require the following:

- An Azure subscription that can create and delete resources in the subscription. If you do not have an Azure subscription, you can create a free Azure account at <https://azure.microsoft.com/free>.
- Access to an internet browser to log into the Azure portal: <https://portal.azure.com>.
- Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Resource tags

Resource tags are used to provide *metadata* or *descriptive information* for Azure resources; metadata is a way to describe data; think of it like a sticky note, comments in a document, or a tooltip – a sticky label that provides further information on the object it is describing. This is why it's called a *tag*.

Resource tags can be created via the Azure portal, PowerShell, the Azure CLI, ARM templates, or the REST API; they can also be managed via Azure Policy (which we will cover in the *Azure Policy* section of this chapter). Up to 15 resource tags for each resource can be created, and there is no automatic inheritance by resources; if a tag is set at the resource group level, the resource tag only applies to the resource it is attached to; this may be useful if we want to group things logically by a resource group, but have a way to independently label the resource with metadata that is not tied to a resource group or subscription.

Each resource tag consists of a tag name and tag value, which forms a *key-value* pair; you can define the name and value as you wish. This can be seen in the following diagram:

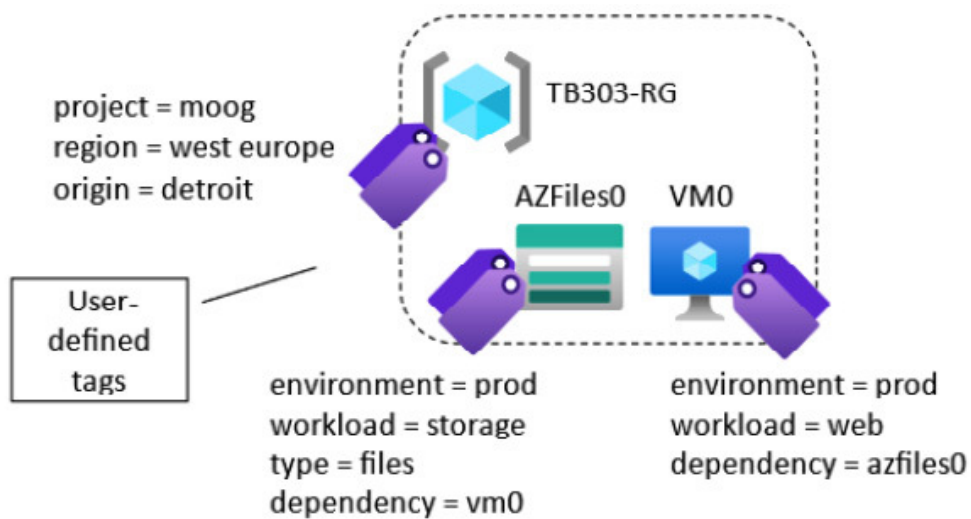


Figure 9.1 – Resource tags

This data can then be used to logically organize all the resources that share the same values; resource tags can be used to take action or enforce a policy on all

resources that share the same resource tags or are flagged with the same values. The following are some common use cases for tags:

- **Billing information:** Cost center, billing ID, project ID, team, business unit, region, and so on
- **Ownership information:** Department or team, named point of contact, stakeholder, and so on
- **Purpose information:** Environment (dev, test, stage, or prod), application name, and so on
- **Classification and compliance information:** SLA, confidentiality marking, region, compliance standard, and so on

One of the most common use cases for tags will be to capture *billing information* for cost management purposes; a cost center/project/team tag could be used where you wish to use a show-back, charge-back for granular cost reporting for resources consumed within a subscription. Resource tags appear in the CSV export of an Azure bill so that the resource tag column can be filtered on and used with automation.

In this section, we looked at tagging Azure resources to create a taxonomy for governance. The following section looks at resource locks.

Resource locks

Resource locks are used to prevent resources from being modified, but more importantly, they are used to prevent resources from being accidentally deleted; locks *override* any permissions that have been set through RBAC.

Resource locks are managed at the subscription, resource group, and resource level and can be one of the following types:

- **Read-only lock:** Admins cannot delete or update a resource.
- **CanNotDelete lock:** Admins can update a resource but not delete one.

Unlike resource tags, resource locks are inherited by child resources. This means that all the resources in that scope will inherit a parent scope lock. You can add both lock types to resources; multiple locks could be applied to a resource, with the most restrictive inherited lock applying and taking precedence. The following diagram aims to visualize the levels that locks can be applied and inherited at:

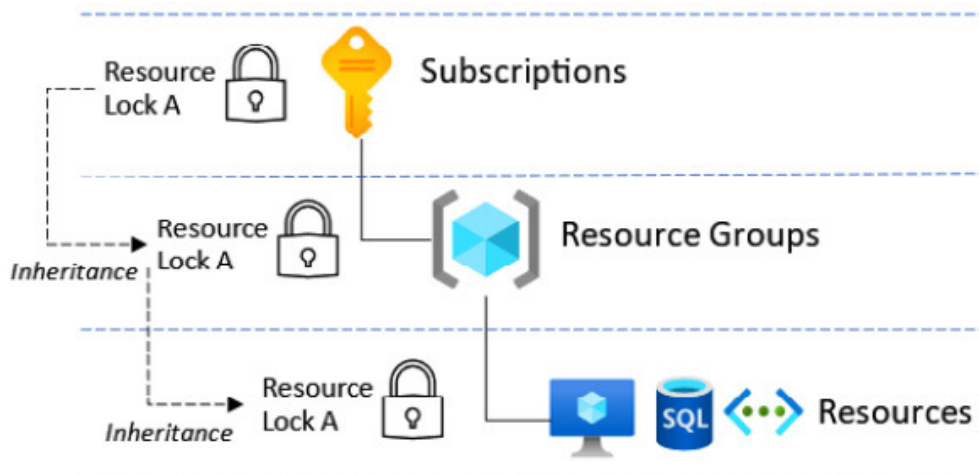


Figure 9.2 – Azure resource locks

Resource locks are applied through the Azure portal, Azure PowerShell, the Azure CLI, ARM templates, or the REST API. You must be an *owner* of a resource or have the *User Access Administrator* role to create a resource lock; alternatively, a custom role can be used that grants the right to create or delete locks.

In this section, we looked at resource locks and how they can be used. The following section looks at RBAC.

Role-based access control

We learned about the basic concepts of **RBAC** in [Chapter 3, Core Azure Architectural Components](#), when we looked at assigning access to manage Azure subscriptions for billing and resource management.

To recap this functionality provided by Azure AD, RBAC is a concept that refers to authorized user access based on defined roles that have been assigned. It allows you to create granular access control to Azure resources through defined roles, as well as through custom roles, and you can segregate duties by granting only the access that's required to perform the required tasks.

It is a good practice for governance to only allow the minimum access required to complete a task. This is the basis for the principle of least privilege and should always be adopted. So, users are only given access through a role(s) that's the most appropriate for the tasks they need to carry out.

This enhances governance and control of user access management as the permissions are not given directly to individuals or groups but given to roles; each role has a set of associated permissions. When a user or group is assigned that role, they receive the role's associated permissions.

It is a good practice not to assign individual users to RBAC but to add users to groups and then assign RBAC to the groups; it is easier for governance and control group membership this way.

Roles are specified at what level they apply and are inherited from the parent level. The following diagram visualizes this approach:

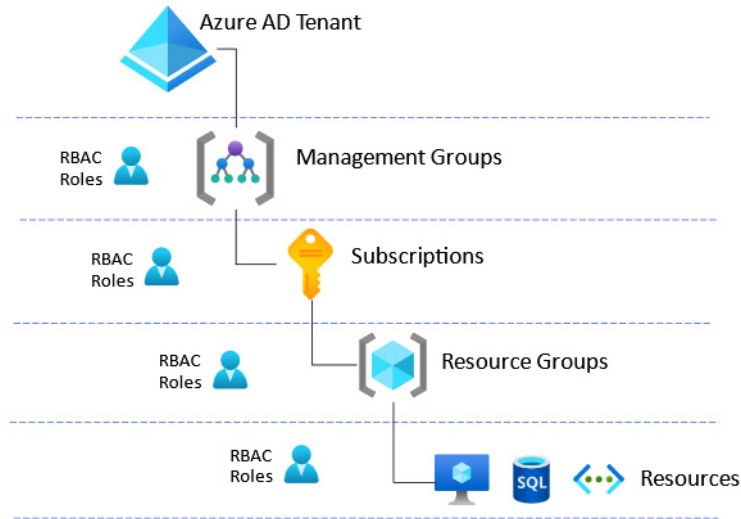


Figure 9.3 – RBAC scope level

RBAC assignment is managed via the **Access Control (IAM)** blade in the portal for each resource you wish to control access over. RBAC is based on four elements, as follows:

- **Security principal:** This represents an *identity*; this can be *User*, *Group*, *Service Principal*, or *Managed Identity*.
- **Role (definition):** This represents a collection of permissions the security principal will receive; what actions they can take on the resource, such as delete, write, and so on. There are several built-in roles, and you can create custom roles, as we referenced earlier in this chapter.
- **Scope:** This represents the resource level that this access will apply to. The scopes from the broadest to narrowest are *management group*, *subscription*, *resource group*, and *resource*; this is structured in a parent-child relationship. A good practice is not to set the scope too wide and only at the least level needed.
- **Role assignments:** This represents the process of attaching a *role definition* to a *security principal* to provide access; creating a role assignment grants access and removing a role assignment revokes access.

The following are the three core RBAC roles for controlling access to Azure resources:

- **Owner:** This role has full access to resources. In addition, it can assign access to others.
- **Contributor:** This role is the same as the *Owner* role, apart from that it *cannot* assign access to others.

- **Reader:** This role can only see resources; it cannot create, edit, delete, or manage any resources.

If you feel that the default permissions of a built-in role may provide too much control or be too restrictive, then a more granular custom role can be created with only the permission associated with your required role; these custom roles can then be assigned to *users, groups, and service principals* at the management group, subscription, and resource group scope, the same as for the built-in roles.

Custom roles can be shared among subscriptions within the same tenancy's Azure AD directory; there is a limit of 5,000 custom roles that can be created per directory (Azure China has a limit of 2,000). We will look at creating a custom role in *Exercise 2 – creating a custom RBAC role* in this chapter.

In this section, we looked at granting least privilege access to resources through the RBAC functionality for all Azure resources. The following section looks at Azure Policy.

Azure Policy

Azure Policy is a set of rules for resource creation and management that apply across multiple subscriptions; it defines what actions are allowed within a subscription and assesses resources to ensure that compliance standards are met or enforce organization mandates drift, or that non-compliance can be remediated through automation.

Some typical example use cases of Azure Policy could limit what regions can be accessed for resources to be created so that data sovereignty can be complied with. You can even limit VM types or storage types so that expensive or operationally inefficient resources are not created.

The following are the key differences to understand between *Azure Policy* and *Azure RBAC*:

- *Azure Policy*:
 - Controls what can be done (regardless of the user)
 - Focuses on resource properties
 - Applied to resources

An example of *Azure Policy* would be that you are the contributor of a resource group, but a policy blocks you from deploying a VM in WestUS or you cannot deploy a DS12v2 but can deploy a DS4_v2.

We will look at this in a hands-on exercise in this chapter.

- *Azure RBAC*:
 - Controls who can do what (specific to each user)
 - Focuses on user actions
 - Applied to each action
 - Create, read, update, and delete
 - Built-in and custom roles
 - Default Deny, explicit Allow

An example of *Azure RBAC* would be having contributor rights on a resource group in a dev/test subscription but only VM admin rights in a production subscription.

Policies are described in **JavaScript Object Notation (JSON)** format via *policy definitions*, which are the business rules to be applied to the policy; these *rules (policy definitions)* can be grouped to form a *policy initiative (or policy set)*. The policy definition is assigned to a resource-level scope and applied by *policy assignments*; the policy will *evaluate* all resources in the level assigned.

The following diagram shows that policies are scoped at the subscription, resource group, and resource levels:

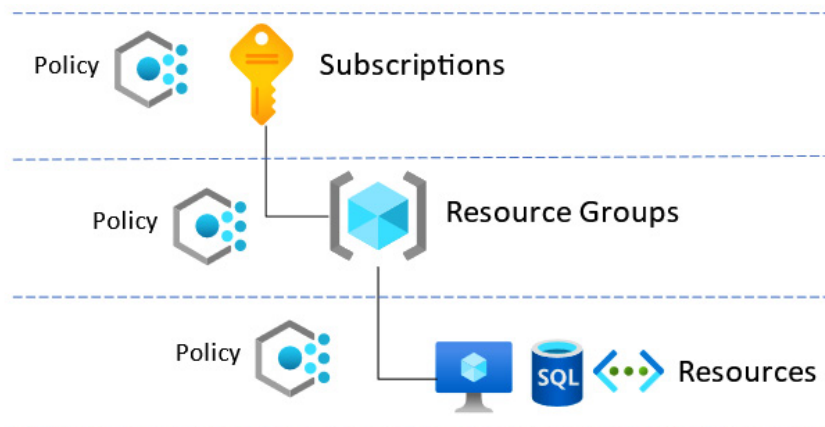


Figure 9.4 – Azure Policy scope level

In this section, we looked at Azure Policy and its differences from RBAC. The following section looks at Azure Blueprints.

Azure Blueprints

Azure Blueprints is much the same as we can think of blueprints outside of a technology discussion; it provides patterns, designs, and definitions for creating something. We can think of this along the lines of a blueprint for a house, a car, a space rocket, and so on; an Azure blueprint is no different from this concept.

An Azure blueprint is a package or representation of a collection of defined, prescribed, repeatable resources to be deployed that conform to an organization's governance standards and patterns when implemented. This allows governance and design parameters to be defined that rapidly allow teams to repeatedly stand up projects and initiatives within the blueprint's control.

Azure blueprints can be created through the Azure portal, Azure PowerShell, the Azure CLI, ARM templates, or the REST API; blueprint definitions are saved to a *management group* or *subscription*, which requires *contributor* access.

Blueprints are represented as objects and are replicated to multiple Azure regions so that they can always be accessed from whichever region you wish to deploy resources to. This allows you to create repeatable environments that use the same defined artifacts each time across all subscriptions.

One blueprint can be assigned to govern several subscriptions, so they always have the same definitions; blueprints are assigned to the management or subscription level.

A blueprint is made up of a *blueprint definition* (what *should* be deployed) and a *blueprint assignment* (where is it being deployed). It is composed of artifacts, and the blueprint can be thought of as the container, package, or scaffolding, as it were. The following diagram aims to visualize the concepts of Azure Blueprints:

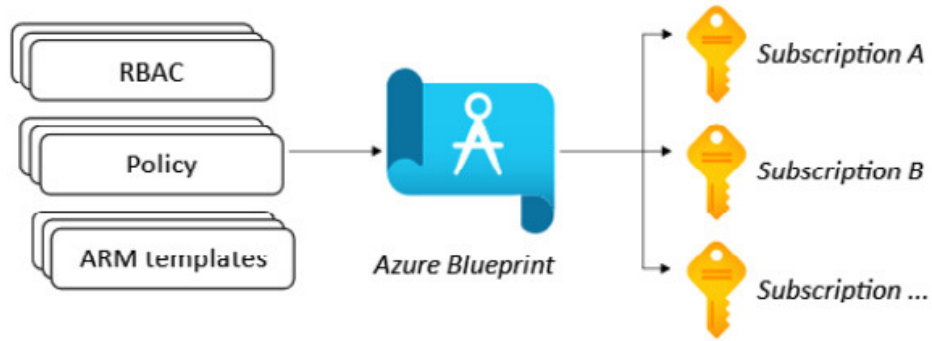


Figure 9.5 – Azure Blueprints

In summary, Azure Blueprints allows governed environments to be quickly and repeatably deployed across subscriptions to a governed resources standard using a set of composed artifacts.

In this section, we looked at Azure Blueprints. In the next section, we'll look at the CAF for Azure.

The Cloud Adoption Framework for Azure

The **CAF** is a collection of proven tools and documentation, including best practices, reference architectures, and implementation guidance. This allows business and technology strategies to be aligned so that they accelerate cloud adoption in a controlled and governed manner. The focus for this content is the *cloud architect*, who is the conduit for discussions and activity between the business and operations teams, and acts as the thought leader for the organization.

The CAF provides various methodologies, as per the following diagram:

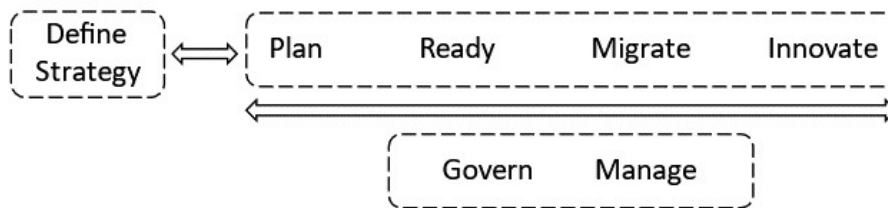


Figure 9.6 – Azure CAF methodologies

Let's look at these in more detail:

- **Strategy:** Define justification and outcomes.
- **Plan:** Align business outcomes to actionable adoption plans.
- **Ready:** Preparation of the cloud environment.
- **Migrate:** Existing workloads move and are modernized.
- **Innovate:** New workload development using cloud-native or hybrid solutions.
- **Govern:** Control of the environment.
- **Manage:** Support and operations management.

The CAF documentation can be accessed via the following URL:

<https://docs.microsoft.com/azure/cloud-adoption-framework>.

In this section, we looked at the CAF for Azure. The following section looks at the hands-on exercises for this chapter, which will help you build on the skills you've learned in this chapter.

Hands-on exercises

To support your learning with some practical skills, we will create some of the resources that were covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – assigning access with RBAC
- Exercise 2 – creating a custom RBAC role
- Exercise 3 – adding a resource lock to a resource group
- Exercise 4 – enabling resource tagging with Azure Policy
- Exercise 5 – limiting the resource creation location with Azure Policy

Getting started

To get started with these hands-on exercises, you will need an Azure subscription that has access to create and delete resources in the subscription; you can use an existing account that you have created as part of the exercises from any chapter in this book. Alternatively, you can create a free Azure account by going to <https://azure.microsoft.com/free>.

This free Azure account provides the following:

- 12 months of free services
- \$200 credit to explore Azure for 30 days
- 25+ services that are always free

Exercise 1 – assigning access with RBAC

This section will assign access to the resource group with RBAC using the Azure portal.

The following subsections cover how to complete this exercise, segregated into tasks for ease of understanding.

Task – accessing the Azure portal

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – creating a new resource group that requires access

2. In the assigning with RBAC:resource group requiring access, creating" search bar, type **resource groups**; click **Resource groups** from the results list.
3. From the **Resource groups** blade, click **+ Create** via the top toolbar.
4. From the **Basics** tab, set the **Project** and **Resource** details as required.
5. Click **Next: Review + create**.
6. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings are as required, you can click **Create**.
7. You will receive a notification that the resource group was created successfully.
8. Click **Go to resource group** from the **Notifications** blade. Alternatively, navigate to the resource group instance.

Task – assigning access to a resource group

9. From the created **Resource groups** blade, click **Access control (IAM)** from the left-hand side menu.
10. From the **Roles** tab, review the built-in roles; under the **Details** column, you can click on **View** to see the permissions that role will grant.
11. From the **Role assignments** tab, review the current assignments.
12. From the **Check access** tab, please review the options for **My Access** and **Check access**, and then click **Add role assignment**.

13. From the **Role** tab, click **Owner** or the role you wish to assign.
14. Click **Next**.
15. For this exercise, leave **Assign access to** set to the default of **User, Group, or Service Principal**.
16. From **Members**, select the members to be assigned the role.
17. Click **Next**.
18. Click **Next: Review + assign**.
19. You will receive a notification that the role assignment was added.
20. Click the **Role assignments** tab from the **Access control (IAM)** blade and review the new assignment that was created.

In this exercise, we successfully created a resource group and assigned the RBAC role to a user for access. In the following exercise, we will look at creating a custom RBAC role.

Exercise 2 – creating a custom RBAC role

This section will look at creating a custom RBAC role.

You can use an existing Azure AD instance to perform this exercise if you wish.

The following subsections cover how to complete this exercise, segregated into tasks for ease of understanding.

Task – accessing the Azure portal

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – creating a custom RBAC role

2. In the search bar, type **subscriptions**; click **Subscriptions** from the results list.
3. From the **Subscriptions** blade, click on your subscription.
4. From the left menu of the **Subscriptions** blade, click **Access Control (IAM)**.
5. From the **Access control** blade, click **+Add**.
6. Then, click **Add custom role**.
7. From the **Create a custom role** blade, on the **basics** tab, enter the following information:
 - **Custom role name**.
 - **Description**.
 - **Baseline permissions**: *leave as the default of Start from scratch*.
8. Click **Next**.
9. From the **Permissions** tab, click **+Add permissions**.
10. From the **Add permissions** blade, you can now search for each permission to add. You can now select the actions you wish the custom role to have. Then, click **Add** for each permission you wish to add.
11. From the **Permissions** tab, click **+Add permissions**.
12. Evaluate whether you need to add any exclude permissions; click **Add** or **Cancel** to return to the main blade to continue.
13. Click **Next: Review + create**.

14. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
15. You will receive a message stating that the new custom RBAC role was created; click **OK** on the message.
16. From the main **Access control (IAM)** blade, you will now be able to search/locate the new custom role that was created from the **Roles** tab.

In this exercise, we successfully created a custom RBAC role. In the following exercise, we will look at applying a resource lock to a resource group.

Exercise 3 – creating resource locks

This section will look at adding a resource lock to a resource group using the Azure portal.

The following subsections show how to complete this exercise, segregated into tasks for ease of understanding.

Task – accessing the Azure portal

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – adding a resource lock to a resource group

2. In the search bar, type **resource groups**; click **Resource groups** from the results list.
3. From the **Resource group** blade, open the resource group you created in the previous exercise. Alternatively, create a new resource group (*using the previous exercise's steps*) or use an existing resource group for this exercise.
4. From the **Resource groups** page, click **Locks** under settings from the left navigation menu.
5. From the **Locks** blade, click **+ Add**.
6. From the **Add lock** screen, enter a lock name as required and select **Delete** as the lock type. Then, add any notes as required.
7. You will now see the created lock appear on the list of locks on the **Locks** blade of the resource group.

Task – testing the lock function

8. From the **Overview** blade of the resource group that the lock was created for, click **Delete resource group** from the top toolbar.
9. From the **Delete** blade, type the resource group's name and click **Delete**.
10. You will receive a notification that the resource group failed to be deleted and that it cannot be deleted as it is locked.

In this exercise, we successfully added a delete resource lock to a resource group and validated its operation. In the following exercise, we will look at resource tagging and Azure Policy.

Exercise 4 – enabling resource tagging with Azure Policy

This section will look at enabling resource tagging with Azure Policy using the Azure portal.

The following subsections show how to complete this exercise, segregated into tasks for ease of understanding.

Task – accessing the Azure portal

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – creating a policy assignment

2. In the search bar, type `policy`; click **Policy** from the results list.
3. From the **Policy** blade, click **Assignments** under **Authoring** via the left navigation menu.
4. Click **Assign policy** from the top toolbar.
5. From the **Policy definition** field, under **Basics** on the **Basics** tab, click the *ellipsis* button on the right-hand side of the text box.
6. From the **Available Definitions** page that appears, in the search box, enter `require a tag`.
7. From the policy definition search results, click **Require a tag on resource groups**.
8. Click **Select**.
9. From the **Parameters** tab, in the **Tag Name** field, enter `Environment` as the text value.
10. Click **Next: Review + create**.
11. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.
12. You will receive a notification, stating that the policy assignment succeeded.

Task – testing the policy function

13. In the search bar, type `resource groups`; click **Resource groups** from the results list.
14. From the **Resource groups** blade, click the **+ Create** button via the top toolbar.
15. From the **Basics** tab, set the **Project** and **Resource** details as required.
16. Click **Next: Review + create**.

17. On the **Review + create** tab, click **Create**.
18. You will receive a notification, stating that the resource group failed to be created; click **View error details**.
19. In the **summary** tab, you will see that the policy disallowed the resource; this is the required expected behavior.
20. From the **Tags** tab, enter the name that was defined in the policy; in our exercise, this text value was **Environment**.
21. For this exercise, enter **Production** as the text value.
22. Click **Next: Review + create**.
23. On the **Review + create** tab, click **Create**.
24. You will receive a notification, stating that the resource group was created successfully this time.
25. Search for **Tags** or navigate to the **Tags** blade in the portal. You will see that your tag has been created. Upon clicking your tag, the page for the tag will show all the resources that have been tagged with it.
26. The final task is to clean up and delete the assigned policy that was created in this exercise.
27. From the **Assignments** blade, locate the assignment to delete from the list. Then, right-click and select **Delete assignment** from the pop-up menu.

In this exercise, we successfully created a policy to deny creating a resource that does not have a tag. In the following exercise, we will look at limiting the resource creation location with Azure Policy.

Exercise 5 – limiting the resource creation location with Azure Policy

This section will look at limiting the resource creation location with Azure Policy using the Azure portal.

The following subsections cover how to complete this exercise, segregated into tasks for ease of understanding.

Task – accessing the Azure portal

1. Log into the Azure portal at <https://portal.azure.com>. Alternatively, you can use the Azure desktop app: <https://portal.azure.com/App/Download>.

Task – removing the policy assignment from the previous exercise

2. Before starting this exercise, if you created the policy assignment for the previous exercise and have not deleted this yet, do so now by performing the following step.
3. From the **Assignments** blade, locate the assignment to delete from the list. Then, right-click and select **Delete assignment** from the pop-up menu.

Task – creating a policy assignment

4. In the search bar, type `policy`; click **Policy** from the results list.
5. From the **Policy** blade, click **Assignments** under **Authoring** on the left navigation menu.
6. Click **Assign policy** from the top toolbar.
7. From the **Policy definition** field, under **Basics** on the **Basics** tab, click the *ellipsis* button on the right-hand side of the text box.
8. From the **Available Definitions** page that appears, in the search box, enter `allowed locations`.
9. From the policy definition search results, click **Allowed locations**.
10. Click **Select**.
11. From the **Parameters** tab, select the allowed locations for resource creation.
12. Click **Next: Review + create**.
13. On the **Review + create** tab, review your settings; you may go back to the previous tabs and make any edits if required. Once you have confirmed your settings, click **Create**.

14. You will receive a notification that the policy assignment succeeded.

Task – testing the policy function

15. In the search bar, type `virtual machines`; click **Virtual machines** from the results list.

16. From the **Virtual machines** blade, click the **+ Create** button via the top toolbar and select **Virtual machine**.

17. From the **Basics** tab, set the **Project** details as required.

18. From the **Instance details** tab, select a **region** that is *NOT* on the allowed location for the policy; *this is so we can test the limits of the region that was set in the policy*.

19. You will receive a notification about policy enforcement stating that *in this example, the region selected does not match that allowed in the policy of a location that resources can be created in:*

```
Policy enforcement. Value does not meet requirements on resource:  
Microsoft.Compute/virtualMachines
```

```
The field 'Location' with the value '(Europe) West Europe' is denied
```

20. To remediate this, from the **Instance details** tab, select a **region** that *IS* in the allowed location for the policy; *this is so we can test the limits of the region that was set in the policy*.

21. You will no longer receive the policy enforcement message and will be allowed to continue with resource creation in the policy allowed location.

22. The final task is to clean up and delete the assigned policy that was created in this exercise; this can be achieved by performing the following step.

23. From the **Assignments** blade, locate the assignment to delete from the list. Then, right-click and select **Delete assignment** from the pop-up menu.

In this exercise, we successfully limited the resource creation location with Azure Policy.

This section covered the hands-on exercises for this chapter. The following section provides a summary of this chapter.

Summary

This chapter covered the AZ-900 Azure Fundamentals exam skills area known as *Describe identity, governance, privacy, and compliance features*.

In this chapter, you learned about the various governance services in Azure, including resource tags, resource locks, RBAC, Azure Policy, Azure Blueprints, and the CAF for Azure.

The next chapter covers Microsoft's core tenets of security, privacy, and compliance.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure fundamentals:

<https://docs.microsoft.com/en-us/learn/certifications/exams/az-900>

- Exam AZ-900: skills outline:

<https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>

- Microsoft Learn: Azure Fundamentals – Describe identity, governance, privacy, and compliance features:

<https://docs.microsoft.com/en-gb/learn/paths/az-900-describe-identity-governance-privacy-compliance-features>

Skills check

Challenge yourself with what you have learned in this chapter:

1. Explain a common use for resources tags.
2. Explain the two types of resource locks.
3. Explain RBAC and its scopes.
4. Explain how Azure Policy differs from RBAC.
5. Explain Azure Blueprints.
6. Describe the CAF for Azure.

Chapter 10: Azure Privacy and Compliance

In [Chapter 9, Azure Governance](#), you learned about the skills that covered resource tags, resource locks, **role-based access control (RBAC)**, Azure Policy, Azure Blueprints, and the Cloud Adoption Framework for Azure.

This chapter will outline the Microsoft core tenets of security, privacy, and compliance.

This chapter aims to provide coverage of the AZ-900 Azure Fundamentals Skills Measured section called *Describe identity, governance, privacy, and compliance features*.

By the end of this chapter, you will have learned the following skills:

- How to describe the Microsoft core tenets of security, privacy, and compliance.
- How to describe the purpose of the Trust Center, Microsoft Privacy Statement, the Product Terms site, **Data Protection Addendum (DPA)**, Azure compliance documentation, and Azure Sovereign Regions (Azure Government cloud services and Azure China cloud services).

To support your learning with some practical skills, we will also explore some of the resources and information covered in this chapter through a hands-on exercise.

The following exercise will be carried out:

- Exercise – exploring Microsoft Trust Center Portal

Technical requirements

To carry out the hands-on exercise in this chapter, you will require the following:

- Access to an internet browser

Core security, privacy, and compliance tenets

As we learned in [Chapter 1, Introduction to Cloud Computing](#), security is a *shared responsibility model*. This means that certain responsibilities transfer to the cloud provider in a cloud environment operating model, while other responsibilities are retained by the customer; you should understand when it is *your* responsibility to provide the appropriate level of security and control, and when it is *not* your responsibility but instead that of the cloud services provider to ensure that their platform is kept *compliant* and your data is kept *private*.

The following security model diagram visually sets out the division or separation of responsibilities between the consumer of the cloud resources and the cloud services provider itself:

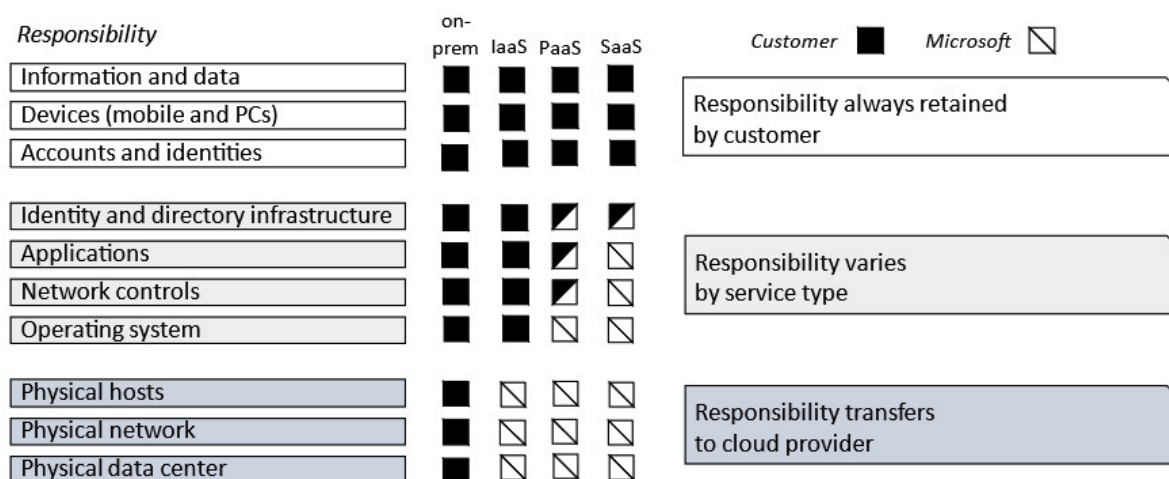


Figure 10.1 – Shared responsibility model

The most critical responsibilities to be aware of are the responsibilities that *you*, as the consumer of cloud services, *always retain* and *your responsibility* to secure and protect.

Security, compliance, privacy, and transparency are fundamental for a *trust model* and are the core tenets of Microsoft Online Services; the following diagram represents Microsoft's trusted cloud principles:

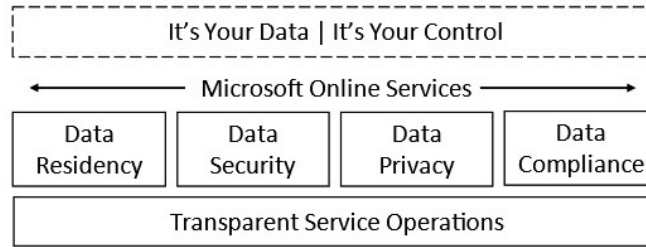


Figure 10.2 – Microsoft trusted cloud principles

The preceding diagram shows that while it is your data and your control, Microsoft is responsible for delivering and operating a cloud services platform that will provide the data residency an organization needs, as well as ensuring it will keep that data *secure*, *private*, and *compliant* with recognized compliance and regulatory standards. These, however, are not just principles, but *contractual guarantees*.

In this section, we looked at Microsoft's trusted cloud principles. The following sections look at how Microsoft delivers on these core tenets.

Trust Center

The **Trust Center** is a publicly accessible web portal that acts as a single point of focus for an organization that needs resources and in-depth information regarding the Microsoft principles of security, privacy, and compliance. The Trust Center can be accessed from <https://www.microsoft.com/trust-center>:

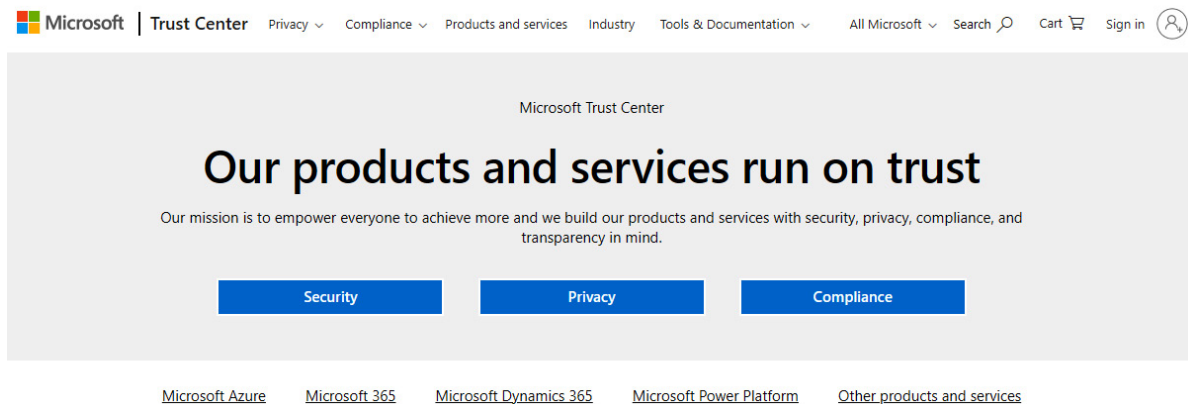


Figure 10.3 – Microsoft Trust Center

The Trust Center is a centralized place for any organization that needs information or resources on security, privacy, and compliance regarding Microsoft Online Services, not just Azure. The following section looks at the *Microsoft Privacy Statement*.

Microsoft Privacy Statement

The **Microsoft Privacy Statement** contains details about how each Microsoft service interacts with your data. It covers how this personal data is *collected*, *the purpose it serves*, and *how it is used*. The Microsoft Privacy Statement extends across all products and services, such as Windows, M365, Azure, and Xbox; across all operating environments such as the cloud and on-premises; and all markets such as commercial, academic, consumer, and so on.

The Microsoft Privacy Statement can be accessed from <https://privacy.microsoft.com/privacystatement>:

Microsoft | Privacy Privacy dashboard Privacy report Privacy resources Privacy Statement

Microsoft Privacy Statement

Last Updated: April 2021 [What's new?](#) [Expand All](#) [Print](#)

Your privacy is important to us. This privacy statement explains the personal data Microsoft processes, how Microsoft processes it, and for what purposes.

Microsoft offers a wide range of products, including server products used to help operate enterprises worldwide, devices you use in your home, software that students use at school, and services developers use to create and host what's next. References to Microsoft products in this statement include Microsoft services, websites, apps, software, servers, and devices.

Please read the product-specific details in this privacy statement, which provide additional relevant information. This statement applies to the interactions Microsoft has with you and the Microsoft products listed below, as well as other Microsoft products that display this statement.

Personal data we collect	Personal data we collect
How we use personal data	
Reasons we share personal data	
How to access and control your personal data	
Cookies and similar technologies	Microsoft collects data from you, through our interactions with you and through our products. You provide some of this data directly, and we get some of it by collecting data about your interactions, use, and experiences with our products. The data we collect depends on the context of your interactions with Microsoft and the choices you make, including your privacy

Figure 10.4 – Microsoft Privacy Statement

In this section, we looked at the Microsoft Privacy Statement and what information it contains. The following section will look at the Microsoft Product Terms site.

The Product Terms site

The **Products Terms** site is an online portal containing the legal agreement and licensing terms and conditions that an organization must comply with through Microsoft commercial licensing programs.

The site covers all products and services; that is, Software and Online Services; these were previously available as separate resources and have now been combined into a single unified online resource.

The Product Terms site can be accessed from

<https://www.microsoft.com/licensing/terms>:

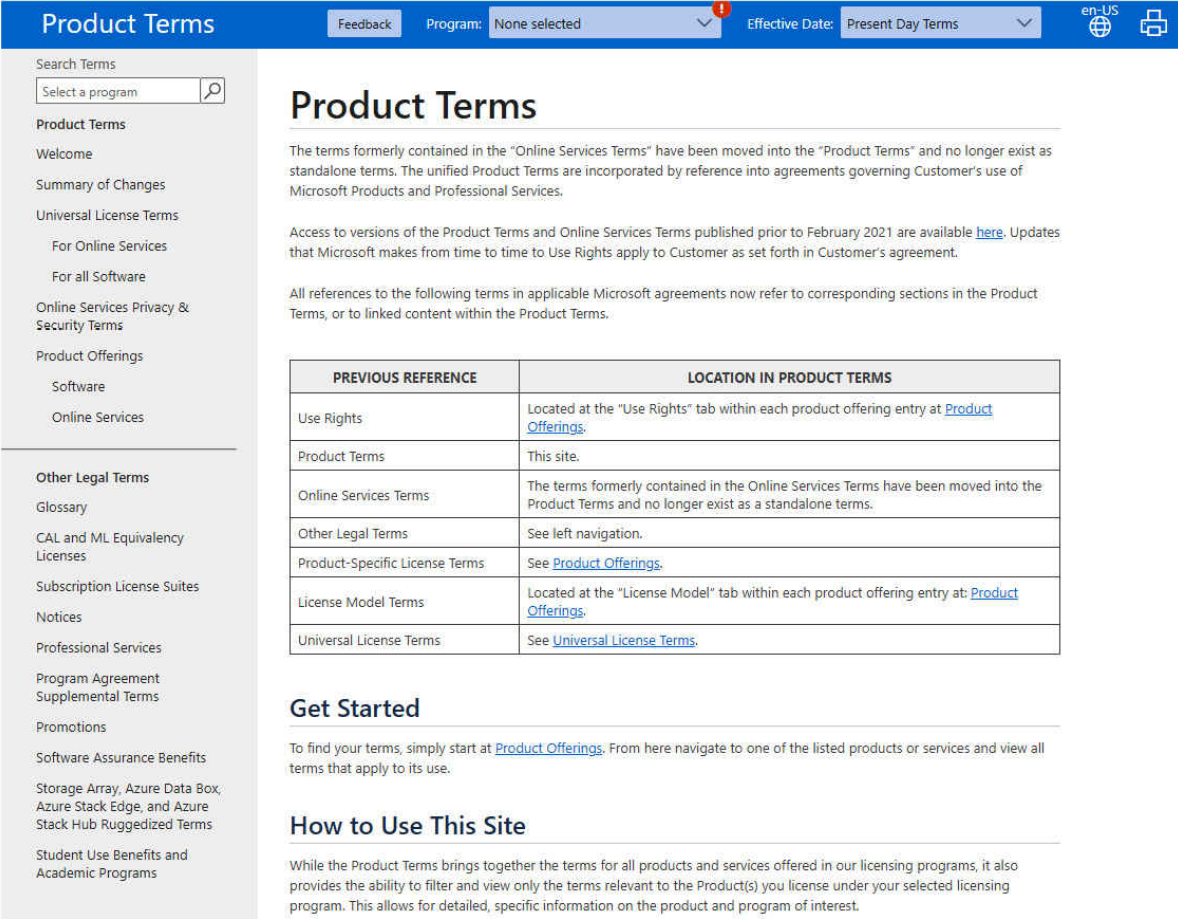


Figure 10.5 – Microsoft Product Terms site

In this section, we looked at the Product Terms site. The following section looks at the Data Protection Addendum.

Data Protection Addendum

The **Data Protection Addendum (DPA)** is an addendum to the *Product Terms site* we looked at in the previous section. It defines the *data processing* and *security terms* for any *Online Services* an organization subscribes to under the Product Terms site.

The current and archived versions of the addendum can be downloaded from the Product Terms site and can also be accessed from

<https://www.microsoftvolumelicensing.com/DocumentSearch.aspx>:

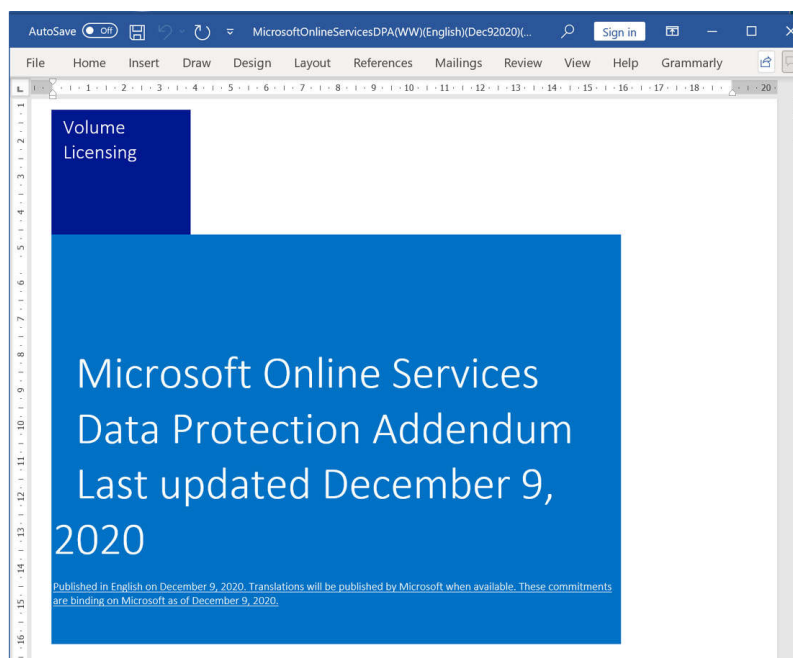


Figure 10.6 – Data Protection Addendum

In this section, we looked at the DPA for subscribed Online Services. The following section looks at the Azure compliance documentation.

Azure compliance documentation

As its name suggests, the **Azure compliance documentation** is an online documentation site that provides detailed information and resources about legal, regulatory standards, as well as compliance an organization has on Azure. The documentation can be accessed at <https://docs.microsoft.com/azure/compliance>:

The screenshot shows the Microsoft Docs website for Azure compliance. The header includes the Microsoft logo, navigation links (Docs, Documentation, Learn, Q&A, Code Samples), a search bar, and a sign-in button. Below the header, there's a blue banner with the title "Azure compliance documentation" and a sub-header "If your organization needs to comply with legal or regulatory standards, start here to learn about compliance in Azure." The main content area is titled "Compliance offerings" and is organized into a grid of 12 categories, each with a list of standards and regulations:

- Global**
 - CIS benchmark
 - CSA STAR Attestation
 - CSA STAR Certification
 - CSA STAR self-assessment
 - SOC 1
 - SOC 2
 - SOC 3
- Global**
 - ISO 20000-1
 - ISO 22301
 - ISO 27001
 - ISO 27017
 - ISO 27018
 - ISO 27701
 - ISO 9001
 - WCAG
- US government**
 - CJIS
 - CNSSI 1253
 - DFARS + CMMC
 - DoD IL2
 - DoD IL4
 - DoD IL5
 - DoD IL6
 - DoE 10 CFR Part 810
 - EAR
 - FedRAMP
- US government**
 - FIPS 140
 - IRS 1075
 - ITAR
 - NDAA Section 889
 - NIST 800-161
 - NIST 800-171
 - NIST 800-53
 - NIST 800-63
 - NIST CSF
 - Section 508 VPATs
- Financial services**
 - 23 NYCRR Part 500 (US)
 - AFM and DNB (Netherlands)
 - AMF and ACPR (France)
 - APRA (Australia)
 - CFTC 1.31 (US)
 - EBA (EU)
 - FCA and PRA (UK)
 - FFIEC (US)
 - FINMA (Switzerland)
- Financial services**
 - FINRA 4511 (US)
 - FISC (Japan)
 - FSA (Denmark)
 - GLBA (US)
 - KNF (Poland)
 - MAS and ABS (Singapore)
 - NBB and FSMA (Belgium)
 - OSFI (Canada)
- Financial services**
 - OSPAR (Singapore)
 - PCI 3DS
 - PCI DSS
 - RBI and IRDAI (India)
 - SEC 17a-4 (US)
 - SEC Regulation SCI (US)
 - SOX (US)
 - TruSight
- Healthcare and life sciences**
 - ASIP HDS (France)
 - EPCS (US)
 - GxP (FDA 21 CFR Part 11)
 - HIPAA (US)
 - HITRUST
 - MARS-E (US)
 - NEN 7510 (Netherlands)
- Automotive, education, energy, media, and telecommunication**
- Regional - Americas**
 - Argentina PDPA
- Regional - Asia Pacific**
 - Australia IRAP
- Regional - EMEA**
 - EU Cloud CoC

Figure 10.7 – Azure compliance documentation

In this section, we looked at the Azure compliance documentation. The following section looks at Azure Sovereign Regions.

Azure Sovereign Regions

Azure supports what is referred to as **Sovereign Regions**; these support greater compliance for specific markets. These regions, as shown in the following diagram, operate isolated instances of the Azure cloud computing platform that run dedicated hardware and isolated networks:

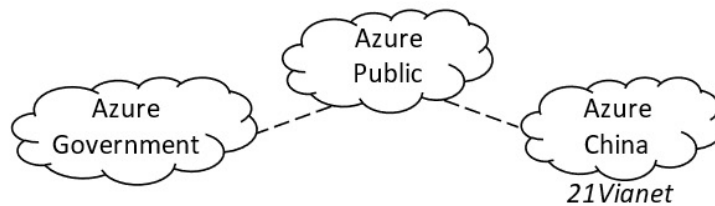


Figure 10.8 – Azure Sovereign Region cloud

As outlined here, the Sovereign Region platforms also have portals with different URLs and service endpoints in DNS:

- **Azure Government:** This is a separate instance of the Azure platform that Microsoft operates; it is for the sole use of US government bodies (and partners):
 - The Azure portal can be accessed via a dedicated URL: <https://portal.azure.us>.
 - The service endpoints to connect to in DNS are in the form of `*.azurewebsites.us`.
 - You can find more information at <https://azure.microsoft.com/global-infrastructure/government>.
- **Azure China (21Vianet):** This is a separate instance of the Azure platform operated by 21Vianet; it is for compliance with Chinese government regulations:
 - The Azure portal can be accessed via a dedicated URL: <https://portal.azure.cn>.
 - The service endpoints to connect to in DNS are in the form of `*.chinacloudsites.cn`.
 - You can find more information at <https://www.azure.cn>.

In this section, we looked at the Azure Sovereign Region clouds. The following section looks at a thought exercise.

Thought exercise

Returning to our online pizza company *MilesBetter Pizza*, they wish to know where they should look to determine whether they need to be compliant with regulatory standards such as **Payment Card Industry Data Security Standard (PCI DSS)** as they handle online transactions. They also don't want to fall foul of any Microsoft product terms for Online Services they have.

In addition, in preparation for an audit, they have been asked to provide evidence in the form of information on the compliance, security, and privacy statements from their cloud service provider of the data stored in Azure. The following diagram visualizes all the resources that are required that were covered in this chapter:

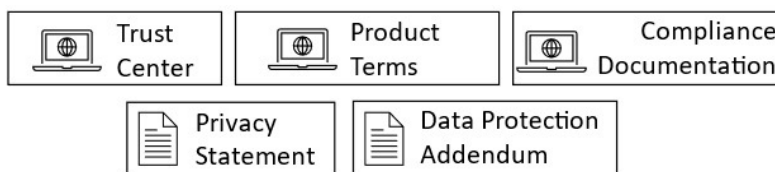


Figure 10.9 – Azure privacy and compliance resources

The following URLs will be required to explore, collate, and present the required audit information:

- Microsoft Trust Center Portal: <https://www.microsoft.com/trust-center>
- Microsoft Privacy Statement: <https://privacy.microsoft.com/privacystatement>
- Product Terms site: <https://www.microsoft.com/licensing/terms>
- DPA: <https://www.microsoftvolumelicensing.com/DocumentSearch.aspx>
- Azure compliance documentation: <https://docs.microsoft.com/azure/compliance>

In this section, we looked at a thought exercise covering privacy and compliance. In the next section, we will complete a hands-on exercise.

Hands-on exercise

To support your learning with some practical skills, we will explore some of the resources and information covered in this chapter.

The following exercise will be carried out:

- Exercise – exploring Microsoft Trust Center Portal.

Getting started

To get started with this hands-on exercise, you will need the following:

- Access to an internet browser

Exercise – exploring Microsoft Trust Center Portal

This section will help you explore the Microsoft Trust Center Portal.

Task – accessing Microsoft Trust Center Portal

1. From a browser, navigate to <https://www.microsoft.com/trust-center>.

Task – exploring the Trust Center Portal

2. The Trust Center Portal can be explored using the *top navigation bar*. The core navigation components to explore can be found in the **Privacy**, **Compliance**, and **Tools & Documentation** drop-down menus. In addition to this, from the **Products and services** navigation component, you can view guidance on security, privacy, compliance, the data's location, GDPR, and more:

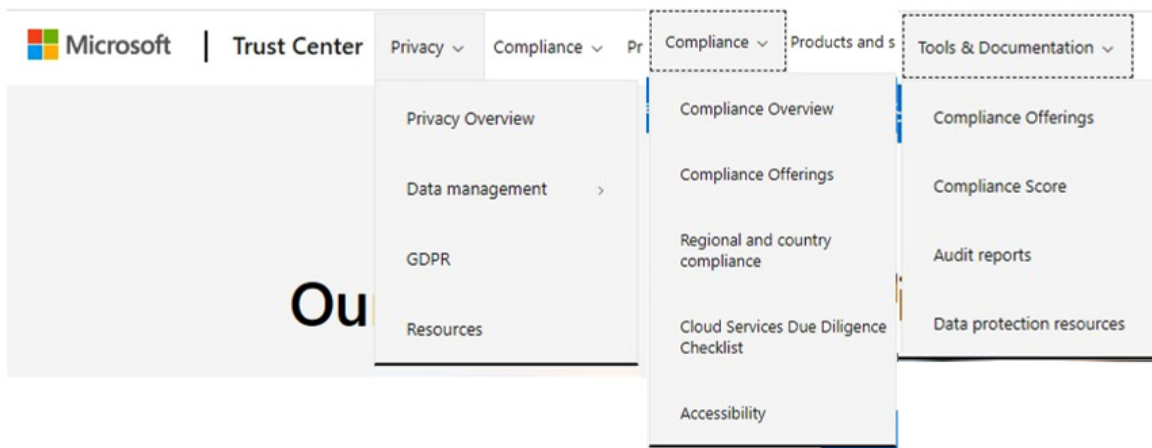


Figure 10.10 – Azure Trust Center Portal

3. From the **Privacy** menu, click **Resources**; among other resources, such as GDR, you will be able to access some of the core privacy resources outlined in the chapter, such as the privacy statement, terms, and data protection addendum. These can be seen in the following screenshot:

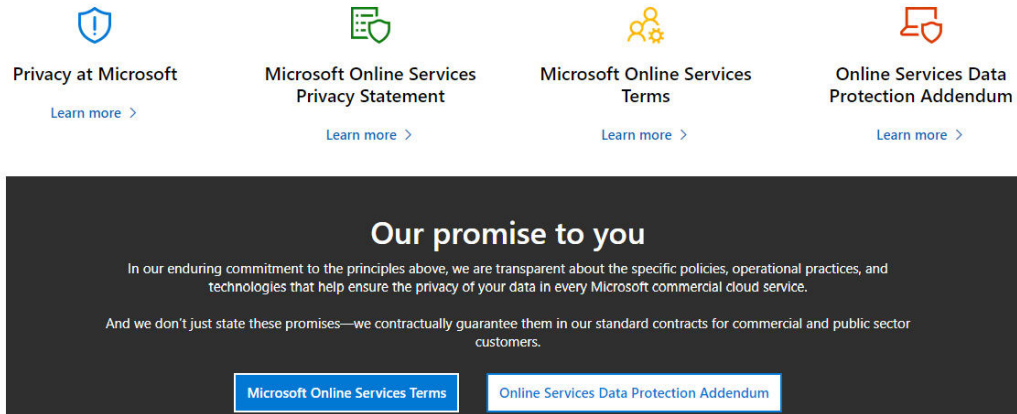


Figure 10.11 – Privacy resources

4. From the **Compliance** menu, you can click through the items to explore each one, such as an overview of compliance, the compliance offerings, regional and country compliance, and so on.
5. From **Products and services**, you can click through to learn about the specific details of each Microsoft product and service.
6. From the **Tools & Documentation** menu, you can click through the items to explore each one, such as audit reports and data protection resources. These can be seen in the following screenshot:

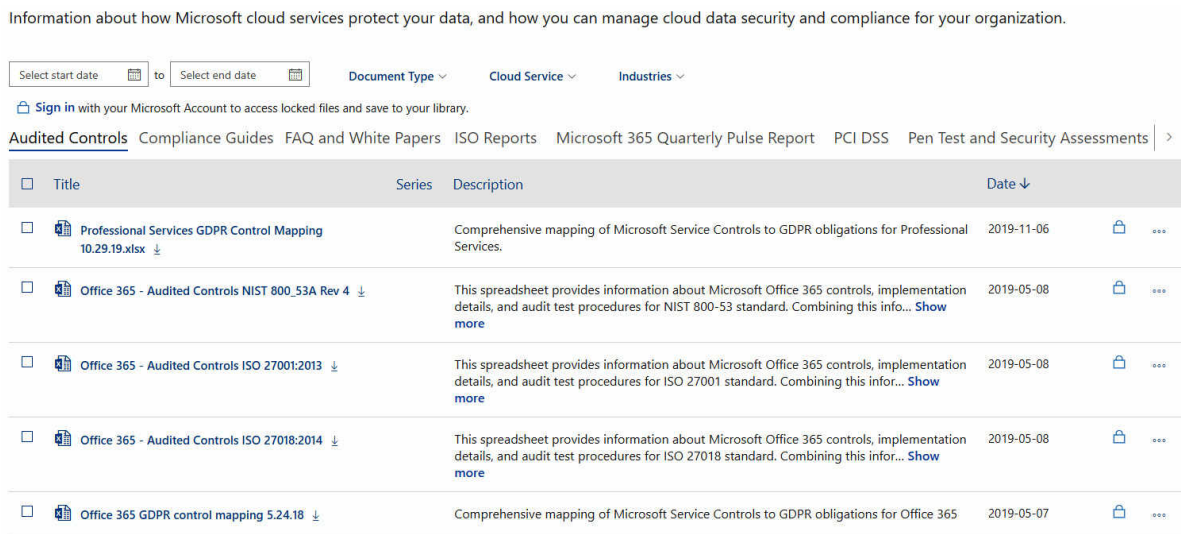


Figure 10.12 – Trust document resources

In this exercise, we explored the Microsoft Trust Center Portal.

Now, let's summarize this chapter.

Summary

This chapter covered some of the AZ-900 Azure Fundamentals exam skills area known as *Describe identity, governance, privacy, and compliance features*.

In this chapter, you learned about the various governance services in Azure, including the core tenets of security, privacy, and compliance. You also learned how to describe the purpose of the Microsoft Privacy Statement, the Product Terms site, **DPA**, Trust Center, Azure compliance documentation, and Azure Sovereign Regions (Azure Government cloud services and Azure China cloud services).

Additional information and study references

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure fundamentals:
<https://docs.microsoft.com/learn/certifications/exams/az-900>
- Exam AZ-900: skills outline: <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- Microsoft Learn: Azure Fundamentals – Describe identity, governance, privacy, and compliance features: <https://docs.microsoft.com/learn/paths/az-900-describe-identity-governance-privacy-compliance-features/>

Skills check

Challenge yourself with what you have learned in this chapter:

1. What are the Microsoft trusted cloud principles?
2. What is the Trust Center?
3. What information is included in the Microsoft Privacy Statement?
4. What is the Product Terms site?
5. What does the DPA cover?
6. Where can you find detailed information on how to conform to regulatory standards such as PCI-DSS?
7. Explain Sovereign Regions.

Section 6: Cost Management and Service-Level Agreements

This section provides complete coverage of the knowledge and skills required for the *skills measured* of the *Describe Azure cost management and service level agreements* section of the exam. We also cover knowledge and skills that go beyond the exam content, so you are prepared for a real-world, day-to-day Azure-focused role.

This part of the book comprises the following chapters:

- [Chapter 11](#), *Azure Cost Planning and Management*
- [Chapter 12](#), *Azure Service-Level Agreements*
- [Chapter 13](#), *Exam Preparation Practice Tests*

Chapter 11: Azure Cost Planning and Management

In [Chapter 10](#), *Azure Privacy and Compliance*, you learned about the skills that covered the Microsoft core tenets of security, privacy, and compliance.

This chapter will cover methods for planning and managing costs.

This chapter aims to provide coverage of the AZ-900 Azure Fundamentals Skills Measured section known as *Describe Azure Cost Management*.

By the end of this chapter, you will have the following skills:

- Be able to understand factors that impact the costs of resources and options for cost control and reduction
- Be able to understand Azure Cost Management
- Be able to understand the Azure Pricing calculator and the **Total Cost of Ownership (TCO)** calculator

To support your learning with some practical skills, we will cover some hands-on exercises while implementing the tools covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – using the Azure Pricing calculator
- Exercise 2 – using the TCO calculator

In addition, this chapter's goal is to take your knowledge beyond the exam's objectives so that you are prepared for a real-world, day-to-day Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will need the following:

- Access to an internet browser.
- A Microsoft account; you can use the same account that you have used for the other exercises in this book. If you do not have a Microsoft account, you can create a free account at <https://account.microsoft.com/account>.

Factors that affect costs

Each Azure *consumption (usage)*-based service has one or more usage meters that define the price rate and unit of cost. Depending on the service, there will be different units of costs.

Billing is performed monthly for each subscription based on resource consumption that's collected from individual meters for that subscription. This means that every month, you may receive a different invoice based on a different set of costs incurred; maybe you consumed more on one resources meter, less on another, and created new resources that created costs against another meter.

The following are primary factors that can affect costs:

- **Purchasing model:** The costs for resources may differ, depending on your purchase model. You can either purchase your Azure directly from Microsoft or through a **Cloud Solution Provider (CSP)**: <https://azure.microsoft.com/pricing/purchase-options>.
- **Resource type:** The costs are specific to your resources; each resource has a billing meter and cost unit. For example, data storage and data transfer will have a unit of billing of GB/month, a VM or Azure SQL database will have a unit of billing of 1 hour, and a premium SSD managed disk will have a unit of billing of 1/month. Storage accounts can charge for any read and write operations unless you're using Premium, in which case these charges are not applicable. It is important to understand the billing units for each resource you create.
- **Location:** The costs will vary between Azure regions.
- **Usage period:** Some resources, such as VMs, can be shut down (de-allocated) to prevent running costs; two identical VMs running for different running hours will have different costs. You would continue to pay for storage costs, but you wouldn't pay for data transfer costs while the VM is not passing network traffic. It is also worth noting that services such as Azure AD Domain Services, Azure Bastion, and the Azure VPN gateway service, once created, will still be billed even if they are not used; the only way to prevent costs for these services is to delete them.
- **Network traffic:** Ingress data transfer (data entering or incoming) for an Azure Region or between resources within the same region is always *free*, but egress data transfer (data leaving or outgoing) from a region is billed at a per-GB unit; this is irrespective of the fact that this is internet traffic or that the region is using a VPN or ExpressRoute circuit.

Note that some resource types are free and have no billing meter or cost implications. The following are some examples of resources that can be created or enabled with no costs. Likewise, removing any of these will not reduce your costs or the invoice you receive:

- User accounts or groups
- Resource groups
- Virtual networks
- Virtual network peering
- Network interfaces
- Network security groups
- Availability sets

It is important to understand what resources have cost implications and what resources don't.

In this section, we looked at factors that affect costs. In the next section, we'll look at how to reduce costs.

Reducing and controlling costs

The following are some of the ways we can reduce and control costs:

- **Optimize resources:** This is an operational activity. Its purpose is to identify any resources that are not used and can be deleted, any resources that can be right-sized onto more cost-optimal resource types or sizes, and identifying any resources that don't need to be running 24/7 and that could be shut down or paused to avoid costs. Any resources running on IaaS should be evaluated to see whether they can be moved to PaaS, serverless, or SaaS. **Azure Advisor** is an essential tool for this activity; tags should also be used to identify costs owners.
- **Azure hybrid benefit:** This is a licensing benefit and allows an organization to maximize any investment in existing on-premises **Software Assurance (SA)**-enabled Windows Server or SQL licenses (or eligible subscription-based licenses); this removes the need to license and pay with the **Pay as You Go (PAYG)** model. For a VM, this does not discount or remove the compute costs or any storage or networking costs; you are still liable for those and need to factor this into the total operating costs of a VM.
- **Azure reservations:** This is a resource benefit and acts as a billing discount mechanism to reduce PAYG consumption charges. It does this by allowing you to commit to paying for an amount of capacity for a fixed term at a discounted rate than you would pay for on the PAYG consumption rate. Reservations are available for a range of resources, such as VMs; they make the most sense and are best used where the workloads must run for long periods or 24/7, where costs are usually reduced by shutting down the VMs to save costs and this is no longer possible. For a VM, this does not apply a discount, remove the software license costs, or any storage or networking costs; you are still liable for those and need to factor this into the total operating costs of a VM.
- **Spot pricing:** This is a resource benefit and allows an organization to make considerable savings based on the ability to take advantage of unused capacity. This is best used for workloads that don't need a specific period in which they must run. This could be tested/dev, analytics, machine learning, batch processes, rendering, and so on.

In this section, we looked at how to reduce and control costs. In the next section, we will look at Azure Cost Management.

Azure Cost Management

Azure Cost Management is provided through a **Cost Management + Billing** dashboard functionality in the Azure portal; it provides core functionality such as cost visibility, optimizations, and accountability.

The following capabilities are provided within the **Cost Management + Billing** function within the Azure portal:

- **Billing:** View and download invoices; view payment methods and make payments.
- **Cost Management:** Perform cost analysis, set cost alerts, and create budgets.

The following screenshot shows the cost analysis screen in the Azure portal:

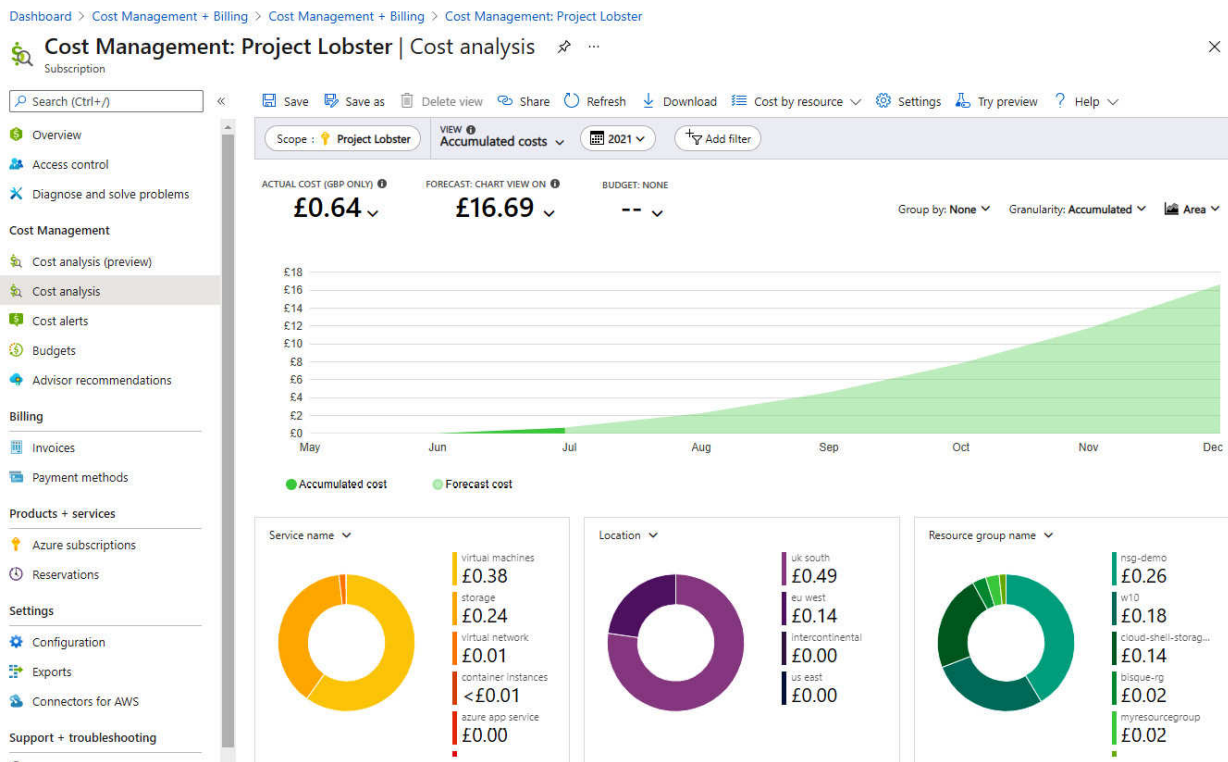


Figure 11.1 – Azure Cost Management

In this section, we looked at Azure Cost Management. In the next section, we will look at the Azure pricing calculator.

Azure Pricing calculator

The **Azure Pricing calculator** is a publicly accessible browser-based tool where you can estimate the cost of services that can be created in Azure.

All Azure resources that can be purchased are displayed in categories that can be browsed through. The calculator has a search function; each resource you can add as an item to the estimate has a hyperlink to the product details for each resource, as well as its pricing page. This is useful if you need to understand the pricing structure for each resource and any factors that may impact costs.

To use the calculator to provide cost estimations for your chosen solution, you must add the required services for your solution to the estimate. Then, you will see a total estimate and breakdown; you can set the currency and then export, save, or share the estimate. Note that the estimates are not intended to be used as actual quotes; the resource's availability, the pricing structure, and its costs may vary from the time of estimation to resource creation.

The Azure Pricing calculator is shown in the following screenshot and can be accessed from <https://azure.microsoft.com/pricing/calculator>:

Figure 11.2 – Azure Pricing calculator

In this section, we looked at the Azure Pricing calculator. In the next section, we will look at the TCO calculator.

TCO calculator

The **TCO** calculator can be accessed as a browser-based tool that can estimate cost savings by moving workloads to Azure. From here, you can generate a report that compares the costs of workloads running in on-premises environments with those running in Azure.

The TCO calculator allows you to enter details for the on-premises workloads. It will then provide some assumptions based on industry average operational costs, including data center facility costs, electricity, labor, hardware, software, and networking costs. It then provides potential costs savings by moving those workloads to Azure. These reports can be saved, downloaded, or shared.

The TCO calculator is shown in the following screenshot and can be accessed from <https://azure.microsoft.com/pricing/tco/calculator>:

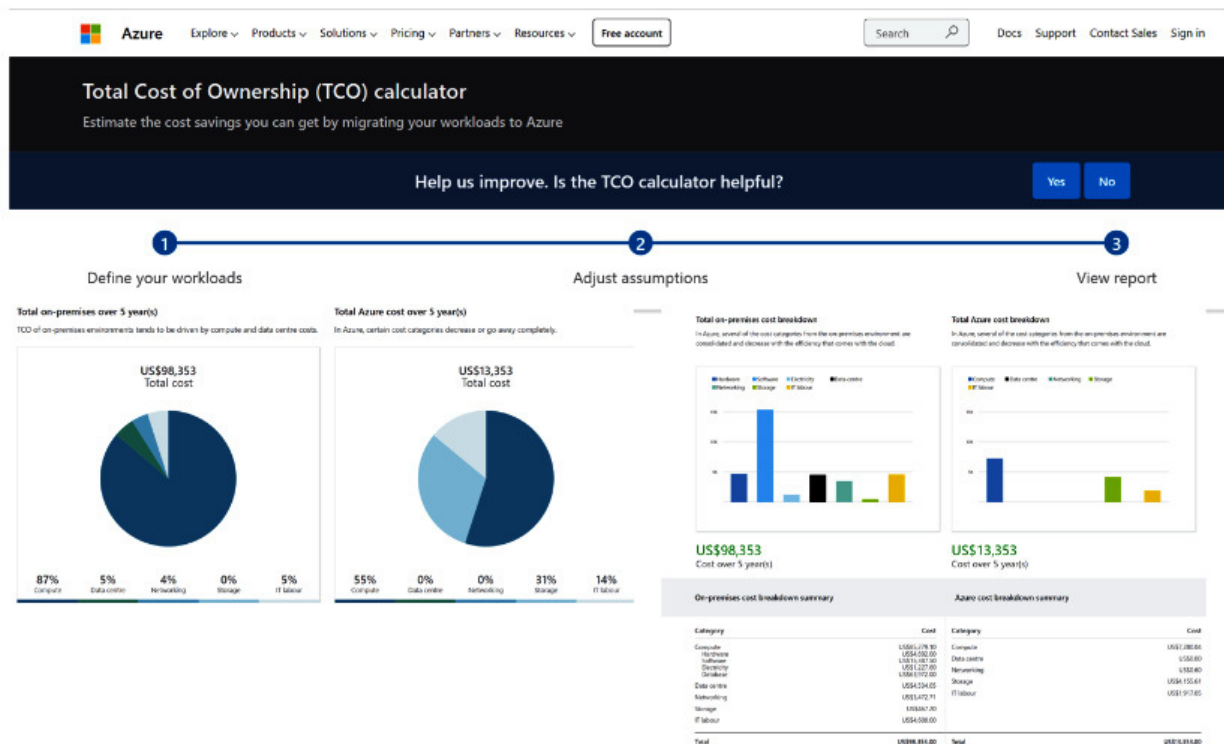


Figure 11.3 – TCO calculator

In this section, we looked at the TCO calculator. In the next section, we will cover some hands-on exercises to help you build on the skills you've learned in this chapter.

Hands-on exercises

To support your learning with some practical skills, we will look at the hands-on use of some of the tools covered in this chapter.

The following exercises will be carried out:

- Exercise 1 – using the Azure Pricing calculator
- Exercise 2 – using the TCO calculator

Getting started

To get started with these hands-on exercises, you will need the following:

- Access to an internet browser.
- A Microsoft account; you can use the same account that you have used for the other exercises in this book. If you do not have a Microsoft account, you can create a free account at <https://account.microsoft.com/account>.

Exercise 1 – using the Azure Pricing calculator

In this exercise, you will create a price estimate with the Azure Pricing calculator; the estimate will be for a simple single-instance Windows VM hosted in the North Europe (Dublin) region.

Task – accessing the Azure Pricing calculator

1. Open a browser and sign into the Azure pricing calculator using your Microsoft account:
<https://azure.microsoft.com/pricing/calculator>.

Task – adding a VM to the estimate

2. From the products tab, click **Virtual Machines**.
3. Scroll down to the VM line item that has been added to the estimate.
4. Adjust the default VM settings to the following for this exercise (or as required):
 - **Region:** North Europe.
 - **Operating System:** Windows.
 - **Type:** (OS only).
 - **Tier:** Standard.
 - **Category:** General purpose.
 - **Instance Series:** Dds v4-series (or as required).
 - **Instance:** D2ds v4 (or as required).
 - **Virtual Machines:** Leave as **qty** of **1** and running for 730 hours.
5. Leave **Savings options** as is.
6. Expand **Managed Disks** and adjust the default settings to the following for this exercise:
 - **Tier:** Premium SSD
 - **Disk Size:** S15: 256 GiB
 - **Disks (qty):** 1
7. Leave **Storage Transactions** as is.
8. Expand **Bandwidth** and adjust the default settings to the following for this exercise:

- **Data transfer type:** Internet egress
- **Source region:** North Europe
- **Routed via:** Microsoft Global Network
- **Outbound data transfer:** 10 GiB

9. Leave **Support** as is.

10. Leave **Programmes and Offers** as is.

11. From the bottom right of the estimate screen, set the currency as required.

Task – saving, exporting, and sharing the estimate

12. From the bottom left of the **Estimate** screen, click **Save as** and enter a name for your estimate.

13. You will see a message stating that the estimate has been saved and that it can be viewed by clicking on the **Saved estimates** tab. Click **Done**.

14. To export the estimate, click **Export**.

15. To share the estimate, click **Share**.

In this exercise, we created an estimate for an Azure resource to be used in a solution using the Azure Pricing calculator. In the next exercise, we will use the TCO calculator.

Exercise 2 – using the TCO calculator

This exercise will create a cost comparison; for example, a typical on-premises environment moving to Azure. You could substitute this with the details of an actual on-premises infrastructure and any workloads you have details of.

Task – accessing the TCO calculator

1. Open a browser and enter the following URL: <https://azure.microsoft.com/pricing/tco/calculator>.

Task – defining your workloads

2. From the **Define your workloads** section, click on the *information* icon next to each selected field.

3. From the **Server** section, select the following information or use your example data:

- Enter a name for the workload or use the provided default.
- **Workload:** Windows/Linux Server.
- **Environment:** Virtual Machines.
- **Operating system:** Windows.
- **Operating system license:** Datacenter.
- **VMs:** 100.
- **Virtualization:** VMware.
- **Cores:** 4.
- **Ram:** 8.
- **Optimize by:** Memory
- **Windows Server 2008/2008 R2:** Off

4. Add any additional server workloads as required.

5. From the **Databases** section, enter the following information or use your example data:

- Enter a name for the database, or use the provided default.
- **Source Database:** Microsoft SQL Server.
- **License:** Standard.
- **Environment:** Virtual Machines.

- **Operating system:** Windows.
- **Operating system license:** Datacenter.
- **VMs:** 15.
- **Virtualization:** VMware.
- **Cores:** 8.
- **RAM:** 32.
- **Optimize by:** Memory.
- **Windows Server 2008/2008 R2:** Off.
- **Destination Service:** SQL Database Managed Instance.
- **Managed instance tier:** General purpose.
- **Managed instance cores:** 8.
- **SQL Server storage:** 100.
- **SQL Server backup:** 100.

6. Add any additional databases as required.

7. From the **Storage** section, enter the following information or use your example data:

- Enter a name for the storage or use the provided default
- **Storage type:** Local Disk/SAN.
- **Disk type:** SSD.
- **Capacity:** 2 TB.
- **Backup:** 2 TB.
- **Archive:** 8 TB.
- **IOPS:** 4,000.

8. Add additional storage as required.

9. From the **Networking** section, enter the following information or use your example data:

- **Outbound bandwidth:** 100 GB

10. Click **Next**.

Task – adjusting your assumptions

11. From the **Adjust assumptions** screen, review all the options and alter them as required. Alternatively, leave the default assumptions as is.
12. Click **Next**.

Task – viewing the report

13. From the **View report** screen, review the cost savings report.
14. You can modify your entries by scrolling to the bottom of the screen and clicking **Back**.
15. You can **download**, **share**, and **save** the report.

In this exercise, we created a cost comparison for an on-premises environment moving to Azure. Now, let's summarize this chapter.

Summary

This chapter covered the AZ-900 Azure Fundamentals exam skills area known as Describe Azure Cost Management.

In this chapter, you learned about cost management and planning.

These skills will have provided you with the confidence to explain and discuss the functionality and usage of the following aspects with a business or technical audience: the factors that impact the cost of resources, such as regions and service types; cost control methods and options for cost optimization and reduction; cost insights and governance through Azure Cost Management; and using the Azure Pricing calculator to estimate resource costs and the **TCO** calculator to calculate the costs of moving workloads to Azure.

In the next chapter, we will cover Azure **Service-Level Agreements (SLAs)**, including their purpose, identifying actions that can impact a SLA, and looking at service life cycles in Azure.

Further reading

This section provides links to additional exam information and study references:

- Exam AZ-900: Microsoft Azure fundamentals:
<https://docs.microsoft.com/learn/certifications/exams/az-900>
- Exam AZ-900: skills outline:
<https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- Microsoft Learn: Azure Fundamentals – Describe Azure cost management and service-level agreements: <https://docs.microsoft.com/learn/paths/az-900-describe-azure-cost-management-service-level-agreements>

Skills check

Challenge yourself with what you have learned in this chapter:

1. What are five factors that can affect costs?
2. List examples of resources with no cost implications; that is, if removed, they would not reduce the monthly bill.
3. What are four ways to reduce and control costs?
4. What are the features of Azure Cost Management?
5. What is the difference between the Azure Pricing calculator and the TCO calculator?

Chapter 12: Azure Service-Level Agreements

In [Chapter 11](#), *Azure Cost Planning and Management*, you learned about the skills that covered identifying factors that can affect costs, options to reduce and control cost, the functionality and usage of the Azure pricing calculator, and the **Total Cost of Ownership (TCO)** calculator.

This chapter will cover Azure **service-level agreements (SLAs)** and the service life cycle.

This chapter aims to provide coverage of the AZ-900 Azure Fundamentals Skills Measured section: *Describe Service-Level Agreements*.

By the end of this chapter, you will be able to do the following:

- Describe the purpose of an Azure SLA.
- Identify actions that can impact an SLA, positively or negatively, such as Availability Zones or composite SLAs.
- Describe the service life cycle in Azure; development, private and public preview, and general availability.

To support your learning with some practical skills, we will also look at some of the resources covered in this chapter through hands-on examples.

The following exercises will be carried out:

- Exercise 1 – exploring the SLA for a service
- Exercise 2 – exploring Azure Preview features

In addition, this chapter's goal is to take your knowledge beyond the exam objectives so that you are prepared for a real-world, day-to-day Azure-focused role.

Technical requirements

To carry out the hands-on labs in this chapter, you will need the following:

- Access to an internet browser.
- A Microsoft account; you can use the same account that you have used for the other exercises in this book. If you do not have a Microsoft account, you can create a free account by going to <https://account.microsoft.com/account>.

Azure SLAs

An *SLA* sets out a customer's expected level of service from their service provider; it can include *responsibilities, vocabulary and terminology, claims and credit processes, service quality, and availability metrics.*

Microsoft defines an SLA as *Microsoft's commitments to uptime and connectivity*, meaning the amount of time the services are online, available, and operational.

Microsoft provides each service with an individual SLA that will detail what is covered by the agreement and any exceptions; a percentage of the monthly fees are credited for any service that does not meet the guarantees. Previews and free services are not provided with an SLA. Information about each service's SLA can be found at the following URLs:

- <https://azure.microsoft.com/support/legal/sla>
- <https://azurecharts.com/sla>

Service availability is expressed as the *uptime percentage over time*; Microsoft SLAs are expressed monthly.

Availability is typically referred to as *9s (nines)*; for example, this can be expressed as *four nines of availability*, meaning the service will be available and fully operational for 99.99% of the defined period. In contrast to availability and uptime, it is also important to consider downtime, which means the amount of time the service will not be available for.

While we see lots of references to *availability* and *uptime* when looking at an SLA that will be provided for a service, the customer and consumer of the services will want to know what that means in the real world and what impact any breach may mean to them. Therefore, it is often the case that the real metric that matters is *downtime*, which means for a given SLA, how long is that service permitted to be down (that is, not available from the service provider)? You should scrutinize any SLA to determine whether that level of downtime is acceptable.

The following table illustrates examples of SLA commitments and downtime permitted per month as part of an SLA:

SLA of a Service	Permitted Downtime Per Month
99.9% (<i>three nines</i>)	43m 49s
99.95%	21m 54s
99.99% (<i>four nines</i>)	4m 22s
99.999% (<i>five nines</i>)	26s

For reference, 99.9% is the minimum SLA that Microsoft provides; 99.999 % is the maximum. It should be noted that 100% can't be provided by Microsoft.

You should also be aware of the concept of a *composite SLA*; this means that when you combine services (such as virtual machines and the underlying services such as storage, networking components, and so on), the overall SLA is lower than the individual highest SLA on one of the services. This is because each service that you add increases the probability of failure and increases complexity. An example exercise will be provided later in this chapter to illustrate this important concept.

The following actions will *positively* impact and *increase* your SLA:

- Using services that provide an SLA (or improve the service SLA), such as *Azure AD Basic and Premium editions* and *Premium SSD managed disks*
- Adding redundant resources, such as *resources to additional/multiple regions*
- Adding availability solutions, such as *using Availability Sets and Availability Zones*

The following actions will *negatively* impact and *decrease* your SLA:

- Adding multiple services due to the nature of *composite SLAs*
- Choosing non-SLA-backed services or free services

The following actions will have no impact on your SLA:

- Adding multiple tenancies
- Adding multiple subscriptions
- Adding multiple admin accounts

The Azure status page (<https://status.azure.com>) provides a global overview of the service health across all regions; this should be the first place you visit, should you suspect there is a wider issue affecting the availability of services globally. From the status page, you can click through to Azure Service Health in the Azure portal, which provides a personalized view of the availability of the services that are being used within your Azure subscriptions.

Service credits are paid through a claims process by a service provider when they do meet the guarantees of the agreed service level. As we mentioned previously, previews and free services are not provided with a financially backed SLA and are not entitled to service credits for any service downtime. You should evaluate all your services to ensure that, where required, you always have an SLA-backed service; as they say, there is often an operational impact that's felt from *free* services.

If you suspect that your services have been affected and that Microsoft has not been able to meet their SLA, then it is your responsibility to take action and pursue credit; you must submit a claim to receive service credit. For most services, you must submit the claim the month after the month the service was impacted. If your services are provided through the Microsoft **Cloud Solution Provider (CSP)** channel, they will pursue this claim on your behalf and provide the service refunds accordingly.

In this section, we looked at Azure SLAs. In the next section, we will look at the Azure service life cycle.

Azure service life cycle

The *service life cycle* defines how each Azure service that's introduced is released and made available.

New Azure services are introduced through *preview services*; these preview services are *not* provided with an SLA or support during the preview (unless there's an explicit service exception).

The service life cycle and the services you can access are as follows:

- **Development:** Not available to the public
- **Private preview:** Available only to a selected audience
- **Public preview:** Available to all customers
- **General availability (GA):** Available to all customers

Preview services can be accessed from the Azure portal; you can get the latest updates on services and their statuses from the following URLs:

- <https://azure.microsoft.com/updates>
- <https://azurecharts.com/presence/rollout>
- <https://azurecharts.com/timeline>

Some preview services are subject to additional terms; you can find this information by going to the *Supplemental Terms of Use for Microsoft Azure Previews* at <https://azure.microsoft.com/support/legal/preview-supplemental-terms>.

While you can use a preview service in production, you should fully evaluate this decision so that you are aware of any limitations of the service and the impact it may have on operations.

You can view the latest Azure updates at <https://azure.microsoft.com/updates>. Here, you can filter to show updates that are *available in preview or development*.

You can keep up to date with the latest service announcements from the Microsoft Azure blog announcements page at <https://azure.microsoft.com/blog/topics/announcements>.

In this section, we looked at the service life cycle in Azure. In the next section, we will look at a thought exercise to conclude this chapter.

Thought exercise

Returning to our digital company, *MilesBetter Pizza*, they have encountered some service availability issues over the last few months. They have noticed that their services have different SLAs, and they need to ensure that, in total, they always meet a target total of 99.9%. However, they are unsure how to calculate and achieve this.

First, the team at *MilesBetter* needs to understand the concept of a composite SLA; as we learned earlier in this chapter, this means that when you combine services, the overall SLA is lower than the individual highest SLA on one of the services. This is because each service that you add increases the probability of failure and increases complexity. The following example illustrates how a composite SLA is calculated across the services:

- Service 1 = 99.9%
- Service 2 = 99.95%
- Service 3 = 99.99%
- Service 4 = 99.99%

The calculation looks as follows, which shows that the target SLA will not be met:

$$0.999 \times 0.9995 \times 0.9999 \times 0.99999 = 0.993 = 99.3\% \text{ total Composite SLA}$$

Therefore, to ensure that they have a total SLA of 99.9%, two of the services must be increased to 99.99% or better, as shown in the following example:

- Service 1 = 99.99% (*action taken to increase from 99.9%*)
- Service 2 = 99.99% (*action taken to increase from 99.95%*)
- Service 3 = 99.99%
- Service 4 = 99.99%

The calculation looks as follows, which shows that the target SLA of 99.99% will now be met:

$$0.9999 \times 0.9999 \times 0.99999 \times 0.99999 = 0.999 = 99.9\% \text{ total SLA}$$

In this section, we looked at the thought exercise for this chapter. In the next section, we will look at the hands-on exercises for this chapter to build on the skills we've

learned so far.

Hands-on exercise

To support your learning with some practical skills, we will use some of the resources we covered in this chapter by performing some hands-on exercises.

The following exercises will be carried out:

- Exercise 1 – exploring the SLA for a service
- Exercise 2 – exploring Azure Preview features

Getting started

To get started with these hands-on exercises, you will need an Azure subscription that can create and delete resources in the subscription. You can use an existing account that you have created as part of the exercises from any chapter in this book.

Alternatively, you can create a free Azure account at <https://azure.microsoft.com/free>.

This free Azure account provides the following:

- 12 months of free services
- \$200 credit to explore Azure for 30 days
- 25+ services that are always free

Exercise 1 – exploring the SLA for a service

This section will look at the resources we can use to gather information on the SLA for a service; we will look at the SLA information for *App Service*, *Virtual Desktop*, *Virtual Machines*, and *Load Balancer*.

The following subsections cover how to complete this exercise. They have been segregated into tasks for ease of understanding.

Task – accessing the SLA summary for Azure services

1. From a browser, go to <https://azure.microsoft.com/support/legal/sla/summary/>:

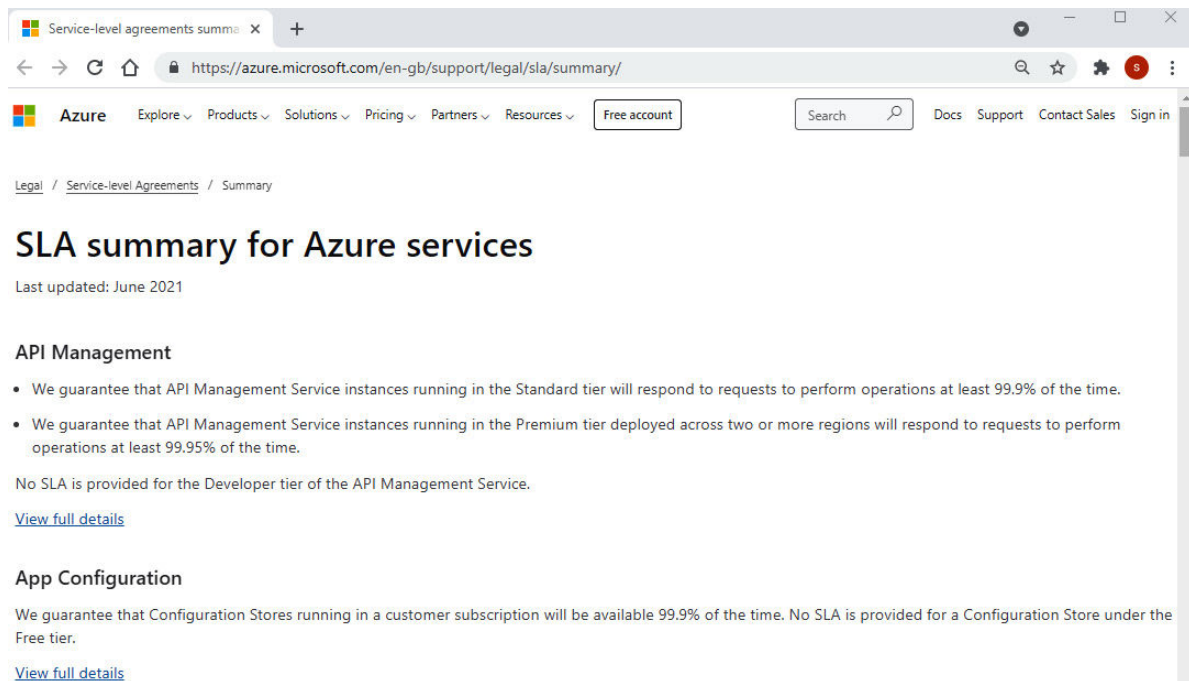


Figure 12.1 – SLA summary for Azure services

Task – viewing the SLA for App Service

2. From the **SLA summary for Azure services** page, scroll down and locate **App Service**.
3. Note what Microsoft specifies they will guarantee and the amount of available time expressed as a percentage; note what doesn't have an SLA.
4. Click on **View full details**. *You can open this in a new tab or window if you prefer.*
5. By looking at the SLA for the individual service, you will see a format that is the same across all services that you should familiarize yourself with.

6. *Expand and explore* the content in the **Introduction**, **General Terms**, and **SLA details** sections:

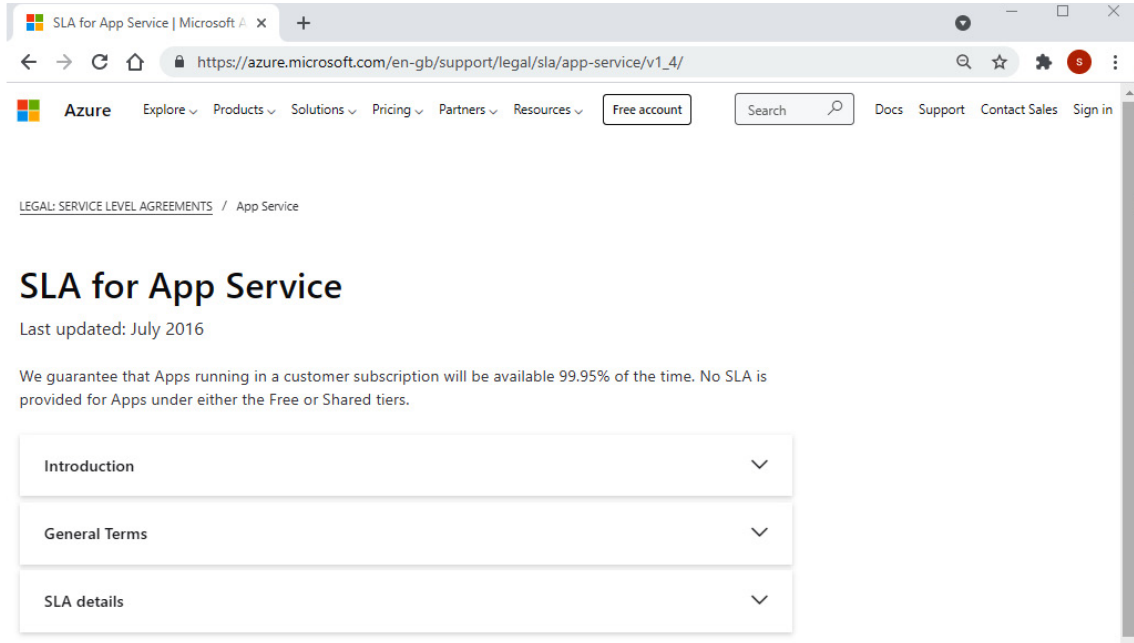


Figure 12.2 – SLA for App Service

7. You can view the **Version History** details of an SLA, **download** the SLA, and **learn more** about the service to go to the product information page for that service.

Task – viewing the SLA for Azure Virtual Desktop

8. From the **SLA summary for Azure services** page, scroll down and locate **Azure Virtual Desktop**.

9. Note that Microsoft does not offer a **financially backed SLA** for this service and that they use the language *...strive to attain at least...*, which means there is no guarantee on the service level available. They note that the Virtual Machine SLA covers the availability of any session hosts.

Task – viewing the SLA for a virtual machine

10. From the **SLA summary for Azure services** page, scroll down and locate **Cloud Services and Virtual Machines**.

11. Note the difference in what is guaranteed for virtual machines that have two or more instances deployed in the same Availability Set and any single-instance virtual machines using premium storage for all disks; ensure you deploy virtual machines in a way that provides the availability and SLA that you need.

12. Click on **View full details**. *You can open this in a new tab or window if you prefer.*

13. *Expand and explore* the content in the **Introduction**, **General Terms**, and **SLA details** sections.

Task – viewing the SLA for Load Balancer

14. From the **SLA summary for Azure services** page, scroll down and locate **Load Balancer**.

15. Note what Microsoft indicates they will guarantee and the amount of time available expressed as a percentage; note what type of Load Balancer SKU does not have an SLA provided.
16. Click on **View full details**. *You can open this in a new tab or window if you prefer.*
17. *Expand and explore* the content in the **Introduction**, **General Terms**, and **SLA details** sections.

In this exercise, we looked at the resources that we can use to gather information on the SLA for a service.

In the next exercise, we will look at where to find information for Azure Preview features.

Exercise 2 – exploring Azure Preview features

In this exercise, we will learn where to find information about Azure Preview features.

The following subsections cover how to complete this exercise. They have been segregated into tasks for ease of better understanding.

Task – exploring the Azure updates site

1. From a browser, go to <https://azure.microsoft.com/updates/?status=inpreview>.
2. From this URL, you can see all the Azure updates that are in preview.

Task – exploring the Azure Preview portal

3. From a browser, go to <https://preview.portal.azure.com>.

From this URL, you can view the preview features for the Azure portal; the title of the page shows **Preview** in brackets so that you know that this is the Preview portal you are exploring. This can be seen in the following screenshot:

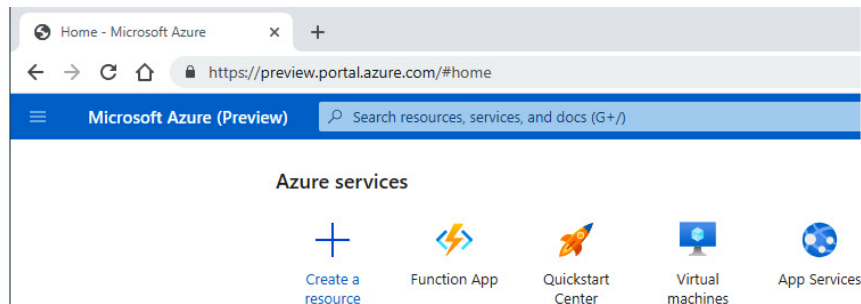


Figure 12.3 – Azure portal – Preview features

Task – exploring Preview features

4. In the search bar, type **preview features** and click **Preview features** from the results list.
5. From the **Preview features** blade, you can explore all the preview features available; you can filter to show only those available for a particular subscription and all the states; that is, if these are not registered or if you have registered them for use.
6. From the top toolbar, you can **Register** for any preview to try out and **unregister** any that you do not wish to be available.
7. You can click on the **Documentation** hyperlink from the **Learn more** column or click on a preview feature from the list, which will open a pop-up blade containing more information about this feature. It will also provide the same hyperlink to the documentation that's relevant to this preview feature;

the **Register** option is also available from this screen. The following screenshot shows the **Previews** features in the Azure portal:

The screenshot displays the Azure portal's 'Preview features' section. At the top, there's a search bar and navigation icons. Below that, the 'Preview features' header includes a 'Refresh' button, a '+ Register' button, and an 'Unregister' button. A filter section allows users to filter by name, subscription (set to 'Test'), state (set to 'All states'), and type (set to 'All types').

The main content area is titled 'Exploring pre-release features with Preview Features' and includes a brief explanation: 'You can now opt in and out of pre-release features with Preview Features. You can access it by searching in Azure or from All Services to see which features are available to your subscription. You can register, unregister, and keep track of registration state here.'

Display name	State	Provider	Release ...	Learn more
<input checked="" type="checkbox"/> On-demand VM guest patching preview	Not registered	Microsoft.Compute	07/11/2019	Documentation
<input type="checkbox"/> Allow ExpressRoute Direct	Not registered	Microsoft.Network	01/06/2018	Documentation
<input type="checkbox"/> VMOrchestratorMultifD	Not registered	Microsoft.Compute	01/03/2020	Documentation
<input type="checkbox"/> VMOrchestratorSingleFD	Not registered	Microsoft.Compute	01/03/2020	Documentation
<input type="checkbox"/> Hotpatch preview	Not registered	Microsoft.Compute	05/03/2020	Documentation
<input type="checkbox"/> Automatic VM guest patching preview	Not registered	Microsoft.Compute	05/03/2020	Documentation
<input type="checkbox"/> AKS Gen2 VM Preview	Not registered	Microsoft.ContainerS...	06/05/2020	Documentation

The right-hand pane shows a detailed view of the 'On-demand VM guest patching preview' feature. It includes the following information:

- Name:** InGuestPatchVMPreview
- State:** Not registered
- Provider:** Microsoft.Compute
- Release date:** 07/11/2019
- Description:** Allows subscription to perform in-guest patching of IaaS VMs

At the bottom of this pane, there are 'Register' and 'Close' buttons.

Figure 12.4 – Azure portal – Preview features

In this exercise, we looked at where to find information about Azure Preview features.

This section covered two hands-on exercises. Now, let's summarize this chapter.

Summary

This chapter covered some of the AZ-900 Azure Fundamentals exam skills in the *Describe Azure cost management and Service Level Agreements* area.

In this chapter, you learned how to describe the purpose of an Azure SLA and the service life cycle in Azure.

The skills you have gained in this chapter should have provided you with the confidence to explain, discuss the functionality, and use the following aspects with a business or technical audience: the purpose of an Azure **SLA**, actions that can impact an SLA either positively or negatively, such as Availability Zones or composite SLAs, and the service life cycle in Azure for development, private and public preview, and general availability.

Then, we concluded this chapter with a thought exercise.

The next chapter will cover exam prep practice tests.

Additional information and study references

This section provides links to additional exam information and study references:

- *Exam AZ-900: Microsoft Azure fundamentals:*
<https://docs.microsoft.com/learn/certifications/exams/az-900>
- *Exam AZ-900: skills outline:* <https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- *Microsoft Learn: Azure Fundamentals – Describe Azure cost management and service level agreements:* <https://docs.microsoft.com/learn/paths/az-900-describe-azure-cost-management-service-level-agreements>

Skills check

Answer the following questions to test what you have learned in this chapter:

1. What is Microsoft's SLA definition?
2. What happens if Microsoft does not meet its SLA for services?
3. What is the minimum SLA Microsoft provides?
4. List two actions that will decrease your SLA.
5. List three different ways to increase your SLA.
6. List two actions that won't increase your SLA.
7. Where can information on each SLA be found?
8. What is a composite SLA?
9. What are the three phases of an Azure service life cycle?
10. When does an Azure service become accessible to the public in the service's life cycle, along with an SLA and support?

Chapter 13: Exam Preparation Practice Tests

In [Chapter 12](#), *Azure Service-Level Agreements*, you learned about the purpose of an Azure **service-level agreement (SLA)**, identifying actions that can impact an SLA, and describing the service life cycle in Azure.

This chapter will cover exam preparation tests in each key skills-measured section from the exam objectives; combined with the hands-on exercises throughout this chapter, you will be confident and ready to apply the knowledge and skills that have prepared you for a real-world, day-to-day Azure-focused role.

Test questions

Each of the following tests maps to the *AZ-900 Microsoft Azure Fundamentals* skills-measured exam outline.

Practice test 1 – cloud concepts

1. In the cloud computing shared responsibility model, who is *a/ways* responsible for the tenant information, data, accounts, and identities?
 - a) The **service provider (SP)**
 - b) The customer
2. Which of the following characteristics relate to the public cloud delivery model?
 - a) Metered pricing and consumption-based billing
 - b) Availability, reliability, fault tolerance, redundancy
 - c) May allow on-premises facilities hosting computing resources to be decommissioned
 - d) Bursting or extending computing resource capacity to a public cloud
3. Which of the following characteristics relate to the private cloud delivery model?
 - a) A dedicated entity (single-tenant) computing model.
 - b) Resources are only available within the capacity provisioned.
 - c) Hardware/ physical resources must be supported; failed hardware must be replaced.
 - d) Provides a choice of location of where on-premises or cloud resources are created.
4. Which of the following characteristics relate to the **platform as a service (PaaS)** service model?
 - a) You have no direct access to the compute layer.
 - b) You have direct access and control of the compute layer, the operating system, and installed software and runtimes.
 - c) You have control over your application, code, and business logic layer; all other layers are provided as a service that you have no access or control over.

5. An Azure **virtual machine (VM)** is an example of the **infrastructure as a service (IaaS)** service type.
- a) True
 - b) False
6. Which are PaaS versions of Microsoft **Structured Query Language (SQL)**?
- a) Azure SQL Server VM
 - b) Azure SQL Database
 - c) Azure SQL Managed Instance
7. An app service is an example of a **software as a service (SaaS)** service type.
- a) True
 - b) False
8. Purchasing on-premises hardware and assets is an example of **capital expenditure (CapEx)**.
- a) True
 - b) False
9. **Operational expenditure (OpEx)** means purchasing on a pay-as-you-go, recurring payment model.
- a) True
 - b) False
10. A consumption-based model means paying only for what you use, and a reservation model means committing to resource usage.
- a) True
 - b) False
11. CapEx is an upfront commitment to purchasing assets.
- a) True
 - b) False
12. Which of the following defines *elasticity*?

- a) The ability to shape the resources needed automatically, burst and scale to meet any peak in demand, and then return to a normal operating baseline
- b) Deploying resources to operate within the required or mandated SLA for those resources

13. Which of the following defines *scalability*?

- a) Increasing resources based on demand, usually in an automated way, triggered upon a metric such as a time or resource threshold being reached
- b) Deploying and configuring resources effectively and efficiently in a short space of time to meet any change in requirements or operational needs

14. Which of the following are *true* or *false* when comparing **disaster recovery (DR)**, **high availability (HA)**, and **backup**?

- a) **HA**—When systems fail and are unavailable, you can run a second instance in the same Azure region.
- b) **DR**—When systems fail and are unavailable, you can run a second instance in another Azure region.
- c) **Backup**—When data is corrupted, deleted, lost, or irretrievable (ransomware), you can restore the instance from another copy of the system.

15. Which of the following protects data from corruption and ransomware?

- a) HA
- b) DR
- c) Backup

Practice test 2 – core Azure services

1. Which solution is recommended to protect against a hardware failure within a data center?
 - a) Availability Set
 - b) **Availability Zone (AZ)**
 - c) Region pair
 - d) Proximity placement group
2. At which management scope can **role-based access control (RBAC)** and *Azure Policy* be targeted?
 - a) Management group
 - b) Subscription
 - c) Resource group
 - d) Resource
 - e) Tag
3. Which limitations apply to management groups?
 - a) The scope is per tenant.
 - b) Only one parent management group is supported and cannot be deleted or moved.
 - c) Management groups can include resource groups or resources.
4. A management group acts as a billing mechanism.
 - a) True
 - b) False
5. Which are mandatory to specify when creating a resource?
 - a) Management group
 - b) Subscription

c) Resource group

6. A subscription can have multiple owners.

a) True

b) False

7. An **Azure Active Directory (Azure AD)**-assigned role gives access to Azure roles.

a) True

b) False

8. The Azure AD Global Administrator role has default access to Azure resources.

a) True

b) False

9. The Contributor role has full access to all Azure resources in a subscription and can create access to others.

a) True

b) False

10. Only internal members of the Azure AD tenant can be given access to Azure resources.

a) True

b) False

11. Which of the following does *not* provide a core Azure platform architecture component functionality provided by **Azure Resource Manager (ARM)**?

a) Creation of resources and dependencies; applying governance and compliance controls

b) Repeatable life cycle deployments; uses template files through **JavaScript Object Notation (JSON)** to define deployment of resources

c) **Identity and Access Management (IAM)**

d) Security posture management

12. Which of the following characteristics relate to resource groups?

- a) Resources must belong in a resource group and can only exist in one resource group, but can be moved between resource groups.
 - b) Resource groups contain metadata about the resources they include.
 - c) Deleting a resource group will remove all resources within that resource group but not delete the subscription or tenant.
13. The **Azure Marketplace** can be used to find details of third-party services and software that have been validated to run in Azure and also create those resources.
- a) True
 - b) False
14. Which of the following requirements will determine using a VM as a compute service?
- a) There is a need to provide control and direct access to the compute layer.
 - b) There are customization requirements, such as customizing the operating system, any software/applications, and runtimes.
 - c) There is a need to extend on-premises computing capacity, maybe for development and testing, DR, and business-continuity scenarios.
15. Which of the following VM types would it be appropriate to select when the workload is memory-intensive and requires a high memory-to-**central processing unit (CPU)** ratio?
- a) D series
 - b) E series
 - c) F series
 - d) N series

Practice test 3 – core solutions and management tools

1. Azure Advisor provides recommendations for which of the following categories?
 - a) Cost, Security, Reliability, Operational Excellence, Performance, Compliance
 - b) Security, Reliability, Operational Excellence, Performance
 - c) Cost, Security, Reliability, Operational Excellence, Performance
2. It is mandatory to implement Azure Advisor recommendations to be supported by Microsoft.
 - a) True
 - b) False
3. Implementing Azure Advisor recommendations will increase an organization's secure score.
 - a) True
 - b) False
4. Azure Advisor recommendations are based on an approach of notification of aspects that are not implemented rather than things that are implemented.
 - a) True
 - b) False
5. The use of deployment template files through JSON for automating the deployment of resources is provided by Azure Policy.
 - a) True
 - b) False
6. Using PowerShell and Bash from Azure Cloud Shell is supported from any device with access to a browser such as Windows, Linux, or macOS.
 - a) True
 - b) False
7. To create a VM from a device that runs Windows, Linux, or macOS, which of the following tools can be used?

- a) Azure portal
 - b) Azure Cloud Shell
 - c) Azure PowerShell (7.x) or PowerShell Core (6.0)
 - d) The Azure **command-line interface (CLI)**
8. Azure mobile apps can be used from an Android or iOS device.
- a) True
 - b) False
9. Azure Functions is an event-triggered serverless compute service PaaS platform.
- a) True
 - b) False
10. Azure Logic Apps is an event-triggered serverless workflow (and orchestration) service PaaS platform.
- a) True
 - b) False
11. Which of the following describes Azure Synapse Analytics?
- a) A Hadoop-based PaaS service used in **machine learning (ML)** scenarios for big data analytics; built on an open source framework
 - b) An Apache Spark-based PaaS service used in ML scenarios for big data analytics; built on an open source framework
 - c) A PaaS service that combines a modern data warehouse with an analytics service using relational database technology; built on **massively parallel processing (MPP)** relational database technology
 - d) A PaaS-managed relational database service; built on SQL Server
12. Which of the following describes Azure Bot Service?
- a) A conversational **artificial intelligence (AI)** service that uses AI to learn human activity and behaviors and interact through speech and text
 - b) A pre-built AI service

c) A custom AI service that provides the ability to make predictions from trained data

13. Which of the following describes Azure IoT Central?

a) A PaaS cloud-hosted Azure solution that collects insights from virtually any number of sensor-fitted devices at a massive scale and takes a building-block approach for customized and complex **Internet of Things (IoT)** scenarios

b) A fully managed SaaS quickstart IoT solution

c) An **end-to-end (E2E)** IoT security solution for the security of telemetry data and communications

14. Which of the following describes Azure DevOps?

a) An integrated code development, version control, and deployment platform

b) A code-hosting platform (*repository*) for open source software, provided as an online-hosted SaaS tool

c) An environment for automating the creation of lab resources from preconfigured base items or **Azure Resource Manager (ARM)** templates while utilizing the fewest administrative resources and the least effort

15. Which of the following describes Azure Monitor?

a) Provides advice on optimizing your Azure resources, such as cost, performance, security, reliability, and operational excellence recommendations.

b) Provides actionable insights into the health, availability, and performance of Azure and on-premises environments by collecting and analyzing logs and metrics.

c) Provides a custom personal view of the health of all your Azure resources; it provides guidance and notifications, such as planned maintenance and other advisories on resource health that are specific to you.

Practice test 4 – security features

1. Which of the following are ways to provide control for the ports of a VM and allow connectivity from the internet on a chosen network port and protocol such as **Remote Desktop Protocol (RDP)** (3389), **Secure Shell (SSH)** (22), **HyperText Transfer Protocol (HTTP)** (80), or **HTTP Secure (HTTPS)** (443)?
 - a) Add a rule to a **network security group (NSG)**.
 - b) Add a rule to an Azure firewall.
 - c) Add a rule to a third-party vendor's **network virtual appliance (NVA)**.
 - d) Azure Traffic Manager.
2. Which of the following are ways to control the ports of a VM and filter traffic between resources within Azure virtual networks?
 - a) Implement an NSG.
 - b) Implement an Azure firewall.
 - c) Implement a third-party vendor's NVA.
 - d) Implement a **virtual private network (VPN)**.
3. An NSG can be used to control access and filter traffic for resources across multiple regions or virtual networks.
 - a) True
 - b) False
4. **Azure Multi-Factor Authentication (Azure MFA)** can be used to encrypt credentials used in code or during the deployment of a VM.
 - a) True
 - b) False
5. Which of the following describes Azure Sentinel?
 - a) A cloud-based centralized solution for storing and managing sensitive information used by an application, service, or resource in an encrypted format, such as passwords, certificates, or keys.

b) A **Cloud Security Posture Management (CSPM)** tool providing security policy and regulatory compliance management and reports, actionable security-hardening tasks, and secure scores that can be used for Azure and on-premises resources.

c) A cloud-based **Security Information and Event Management (SIEM)** and **Security Orchestration, Automation, and Response (SOAR)** tool; it provides security event and log data aggregation, security event threat analysis, and response across public cloud and on-premises environments, such as VMs and network appliances or identity services (AD).

6. Azure DDoS is a cloud-based and Microsoft-managed network security service; it provides protection from network and application attacks that attempt to make a network or application (or any workload) unavailable by flooding it with requests and attempting to exhaust its resources; in addition, it provides attack-analytics and attack-metrics reporting.

a) True

b) False

7. You can associate an NSG with a virtual network in the same region as the VM it will be used with.

a) True

b) False

8. **Defense in depth (DiD)** uses an approach of *never trust, always verify*; this is a concept of thinking beyond traditional network perimeter-based security and adopting a holistic approach to security.

a) True

b) False

9. Zero trust refers to a strategy that places multiple layers of different forms of defenses between attackers and the resources you are trying to protect.

a) True

b) False

10. Azure Dedicated Host is a service that provides physical virtualization hosts for individual customers that are not shared with other tenants to host their Azure VMs for Windows and

Linux workloads.

a) True

b) False

Practice test 5 – identity, governance, privacy, compliance

1. An organization wishes to implement a hybrid identity to allow users to access cloud resources using their AD credentials; they configure Azure AD Connect.

Does this meet their requirement?

2. An organization wishes to decommission its on-premises resources but allow users to access cloud resources using their existing AD credentials; they configure Azure AD Connect.

Does this meet their requirement?

3. Which of the following describes the process of *authentication*?

- a) The process of establishing the identity of a person (or service), and proving they are who they say they are
- b) The process of establishing the level of access a person (or service) has to a resource—that is, what they can access and which actions they may perform

4. Azure AD is a Microsoft SaaS identity service and does not require domain controllers on VMs.

- a) True
- b) False

5. Which of the following devices can be domain-joined to Azure AD?

- a) Windows 10 devices
- b) macOS devices
- c) Android devices

6. For MFA, which of the following authentication methods are available?

- a) Password
- b) Code
- c) Biometrics

7. RBAC is a concept that refers to an authorized user's access based on their assignment of defined roles and provides the ability to create granular access control to Azure resources

through defined roles as well as through custom roles; it allows for **segregation of duties (SoD)**, granting only the access required to perform required tasks.

a) True

b) False

8. Which of the following describes Azure Policy?

a) A package or representation of a collection of defined, prescribed repeatable resources to be deployed, which conform to an organization's governance standards and patterns when implemented; it allows governance and design parameters to be defined that rapidly allow teams to initiate projects and initiatives within the blueprint controls.

b) Allows authorized users access based on their assignment of defined roles and allows them to create granular access control to Azure resources through defined roles and custom roles; it allows for SoD, granting only the access required to perform required tasks.

c) A set of rules for resource creation and management that apply across multiple subscriptions; defines which actions are allowed within a subscription, assesses resources to ensure that compliance standards are met or enforces an organization's mandates drift, or non-compliance can be remediated through automation.

9. Resource tags provide metadata or descriptive information for Azure resources; metadata is a method to describe data.

a) True

b) False

10. Resource locks are used to prevent modification to resources, but more importantly, they are used to prevent the accidental deletion of resources.

a) True

b) False

11. Azure compliance documentation is a collection of proven tools, best practices, reference architectures, implementation guidance, and business and technology strategies to accelerate cloud adoption in a controlled and governed manner.

a) True

b) False

12. The Azure Government region is for the sole use of **United States (US)** government bodies (and partners).

a) True

b) False

13. Azure China is operated by Microsoft.

a) True

b) False

14. Which of the following should be used to assess and report whether an Azure environment meets regulatory requirements?

a) Azure Security Center

b) Microsoft Trust Center

c) Azure compliance documentation

15. Which of the following contains details about how each Microsoft service interacts with your personal data? It covers how this personal data is *collected, the purpose it serves, and how it is used*.

a) Microsoft privacy statement

b) Product Terms site

c) **Data Protection Addendum (DPA)**

Practice test 6 – cost management, SLA, and service life cycle

1. Which of the following will reduce costs for an Azure subscription?
 - a) VMs that are in a stopped (deallocated) state
 - b) Reducing hours any service is running; pausing any services that support this feature
 - c) Applying hybrid benefits to resources
 - d) Applying reservations to resources
2. Which of the following will not reduce costs for an Azure subscription?
 - a) Resize the service tier or resource type.
 - b) Remove unused resources such as network interfaces, NSGs, resource groups, and storage accounts.
 - c) Remove unused resources such as public **Internet Protocol (IP)** addresses, standard load balancers, and **network address translation (NAT)** gateways.
 - d) Reduce traffic going into a region (ingress) from the internet, on-premises, another region, or the same region.
3. Which of the following will impact costs for an Azure subscription?
 - a) Location
 - b) Usage period
 - c) Increase in traffic going into a region (ingress) from the internet, on-premises, another region, or the same region
 - d) Increase in traffic leaving a region (egress) to the internet, on-premises, another region, or the same region
4. Azure Cost Management allows an organization to create budgets, set alerts, and perform cost analysis.
 - a) True

b) False

5. You can create different billing provider mechanisms by creating multiple subscription types.

a) True

b) False

6. Which of the following describes the Azure calculator?

a) A publicly accessible browser-based tool; its purpose is to allow cost estimations of services that can be created in Azure.

b) A publicly accessible browser-based tool that can estimate cost savings by moving workloads to Azure; allows a report to be generated that compares the costs of workloads running in on-premises environments to running in Azure.

c) A dashboard functionality in the Azure portal that provides functionality such as cost visibility, consumption, cost analysis, setting cost alerts, and creating budgets.

7. Services in public preview and those with **general availability (GA)** have an SLA.

a) True

b) False

8. Which of the following phases of the service life cycle are available to all customers?

a) Development

b) Private preview

c) Public preview

d) GA

9. Which of the following will positively impact and increase your SLA?

a) Adding multiple services

b) Adding redundant resources, such as resources to additional/multiple regions

c) Adding availability solutions, such as using Availability Sets and AZs

10. Which of the following apply to SLA service credits?

- a) They are paid through a claims process by an SP when they do not meet the guarantees of the agreed service level.
- b) They are available for all services.
- c) They are automatically applied when an SLA is not met.

Test answers

Practice test 1 – cloud concepts

1. **(b)**

2. **(a), (b), (c)**

Answer (d) is a characteristic of the hybrid cloud delivery model.

3. **(a), (b), (c)**

Answer (d) is a characteristic of the hybrid cloud delivery model.

4. **(a)**

Answer (b) is a characteristic of the IaaS cloud delivery model; answer (c) is a characteristic of the serverless/function as a service (FaaS) delivery model.

5. **True**

6. **(b), (c).**

Answer (a) is an IaaS service type.

7. **False**— *an app service is a PaaS service type.*

8. **True**

9. **True**

10. **True**

11. **True**

12. **(a).**

Answer (b) describes HA.

13. **(a).**

Answer (b) describes agility.

14. **All are true**

15. **(c)**

Answers (a) and (b) protect systems, not data.

Practice test 2 – core Azure services

1. (a)

Answer (b) is recommended to protect against a single data center outage in a region; answer (c) is recommended to protect against an entire region failure; answer (d) is recommended where low latency is required between infrastructure components.

2. (a), (b), (c), (d)

Answer (e) is not a management scope; it is a governance control.

3. (a), (b)

Management groups can only contain subscriptions, not resources or resource groups.

4. **False**—*a subscription acts as a billing mechanism.*

5. (b), (c)

6. **True**

7. **False**

Azure AD-assigned roles do not span Azure roles; access must be explicitly given to access Azure resources.

8. **False**

The Azure Global Administrator does not have default access to Azure resources.

9. **False**

Only the Owner role can create access to others.

10. **False**

Only external business-to-business (B2B) and guest-invited users can be given access to Azure resources.

11. (c)

Security Posture Management

12. **(a), (b), (c)**

13. **True**

14. **(a), (b), (c)**

15. **(b)**

Answer (a) is for general-purpose workloads with a balanced CPU-to-memory ratio; answer (c) is for CPU-intensive workloads with a high CPU-to-memory ratio; answer (d) is for graphical workloads where a GPU is required.

Practice test 3 – core solutions and management tools

1. (c)
2. **False**
3. **True**
4. **True**
5. **False**

This statement describes ARM templates, not Azure Policy.

Azure Policy is a set of rules for resource creation and management that apply across multiple subscriptions.

6. **True**
7. **(a), (b), (c), (d)**
8. **True**
9. **True**
10. **True**
11. (c)

Answer (a) describes Azure HDInsight; answer (b) describes Azure Databricks; answer (d) describes Azure SQL Database.

12. **(a)**

Answer (b) describes Azure Cognitive Services, and answer (c) describes Azure Machine Learning.

13. **(b)**

Answer (a) describes Azure IoT Hub, and answer (c) describes Azure Sphere.

14. **(a)**

Answer (b) describes GitHub, and answer (c) describes Azure DevTest Labs.

15. **(b)**

Answer (a) describes Azure Advisor, and answer (c) describes Azure Service Health.

Practice test 4 – security features

1. (a), (b), (c)

Azure Traffic Manager is a HA solution, not a security solution.

2. (a), (b), (c)

A VPN is a network connectivity solution, not a security solution.

3. **False**

An Azure firewall is required to control access and filter traffic for resources across multiple regions or virtual networks.

4. **False**

An Azure Key Vault instance is required to encrypt credentials used in code or during the deployment of a VM.

5. (c)

Answer (a) describes Azure Key Vault, and answer (b) describes Azure Security Center.

6. **True**

7. **False**

An NSG cannot be associated with the virtual network layer.

You can only associate an NSG with a network interface or a subnet and must be within the same region as the VM it will be used with.

8. **False**

This describes the zero-trust strategy.

9. **False**

This describes a DiD approach.

10. **True**

Practice test 5 – identity, governance, privacy, compliance

1. **Yes**

2. **Yes**

3. **(a)**

Answer (b) describes the authorization process.

4. **True**

5. **(a), (b), (c)**

6. **(a), (b), (c)**

7. **True**

8. **(c)**

Answer (a) describes Azure Blueprints, and answer (b) describes RBAC.

9. **True**

10. **True**

11. **False**

This describes the Azure Microsoft Cloud Adoption Framework (CAF) for Azure.

Azure compliance documentation is an online documentation site that provides detailed information and resources about legal and regulatory standards and compliance on Azure to an organization.

12. **True**

13. **False**

Azure China is operated by 21Vianet; this is for compliance with Chinese government regulations.

14. **(a)**

Answer (b), the Microsoft Trust Center, is a publicly accessible web portal that acts as a single point of focus for an organization that needs resources and in-depth information regarding the Microsoft principles of security, privacy, and compliance; answer (c), Azure compliance documentation, is an online documentation site that provides detailed information and resources about legal and regulatory standards and compliance on Azure to an organization.

15. (a)

Answer (b), the Products Terms site, is an online portal containing a legal agreement and licensing terms and conditions that an organization must comply with through Microsoft commercial licensing programs; answer (c), the DPA, is an addendum to the Product Terms site; it defines security terms and data processing for any online services an organization subscribes to under the Product Terms site.

Practice test 6 – cost management, SLA, and service life cycle

1. (a), (b), (c), (d)

2. (b), (d)

Resources such as network interfaces, NSGs, resource groups, and storage accounts are not billable, and traffic going into a region (ingress) from the internet, on-premises, another region, or the same region is not charged.

3. (a), (b), (d)

Traffic going into a region (ingress) from the internet, on-premises, another region, or the same region is not charged.

4. True

5. True

6. (a)

Answer (b) describes the Azure Total Cost of Ownership (TCO) calculator, and answer (c) describes Azure Cost Management.

7. False

Only services that are GA have an SLA.

8. (b), (c)

9. (b), (c)

Adding multiple services will negatively impact and reduce your SLA; this is due to the nature of compound SLAs.

10. (a)

Previews and free services are not provided with a financially backed SLA and are not entitled to service credits for any service downtime. Claims must be raised by the customer, and there is no automatic claims process for breach of SLA.

Summary

This chapter covered exam preparation tests in each key skills-measured section from the exam objectives.

With the knowledge and skills learned in this book, you will now not only be ready to take the *Azure Fundamentals* certified exam, but through the hands-on exercises throughout this book, you will be confident and ready to apply the knowledge and skills that have prepared you for a real-world, day-to-day, Azure-focused role.

With this being the last chapter in this book, I would like to close off by saying a big thank you to all my readers who have given their time and commitment to broadening their skills and knowledge in Azure through this content; I hope it may act as a springboard and spark your interest to pursue further training and education.

Additional information and study references

Here are some links to additional exam information and study references:

- *Exam AZ-900: Microsoft Azure Fundamentals:*
<https://docs.microsoft.com/learn/certifications/exams/az-900>
- *Exam AZ-900 skills outline:*
<https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE3VwUY>
- *Microsoft Learn: Azure Fundamentals – Describe Azure cost management and service level agreements:* <https://docs.microsoft.com/learn/paths/az-900-describe-azure-cost-management-service-level-agreements/>
- *Microsoft Learn: Learn the business value of Microsoft Azure:*
<https://docs.microsoft.com/learn/paths/learn-business-value-of-azure>

Packt

Packt.com

Subscribe to our online digital library for full access to over 7,000 books and videos, as well as industry leading tools to help you plan your personal development and advance your career. For more information, please visit our website.

Why subscribe?

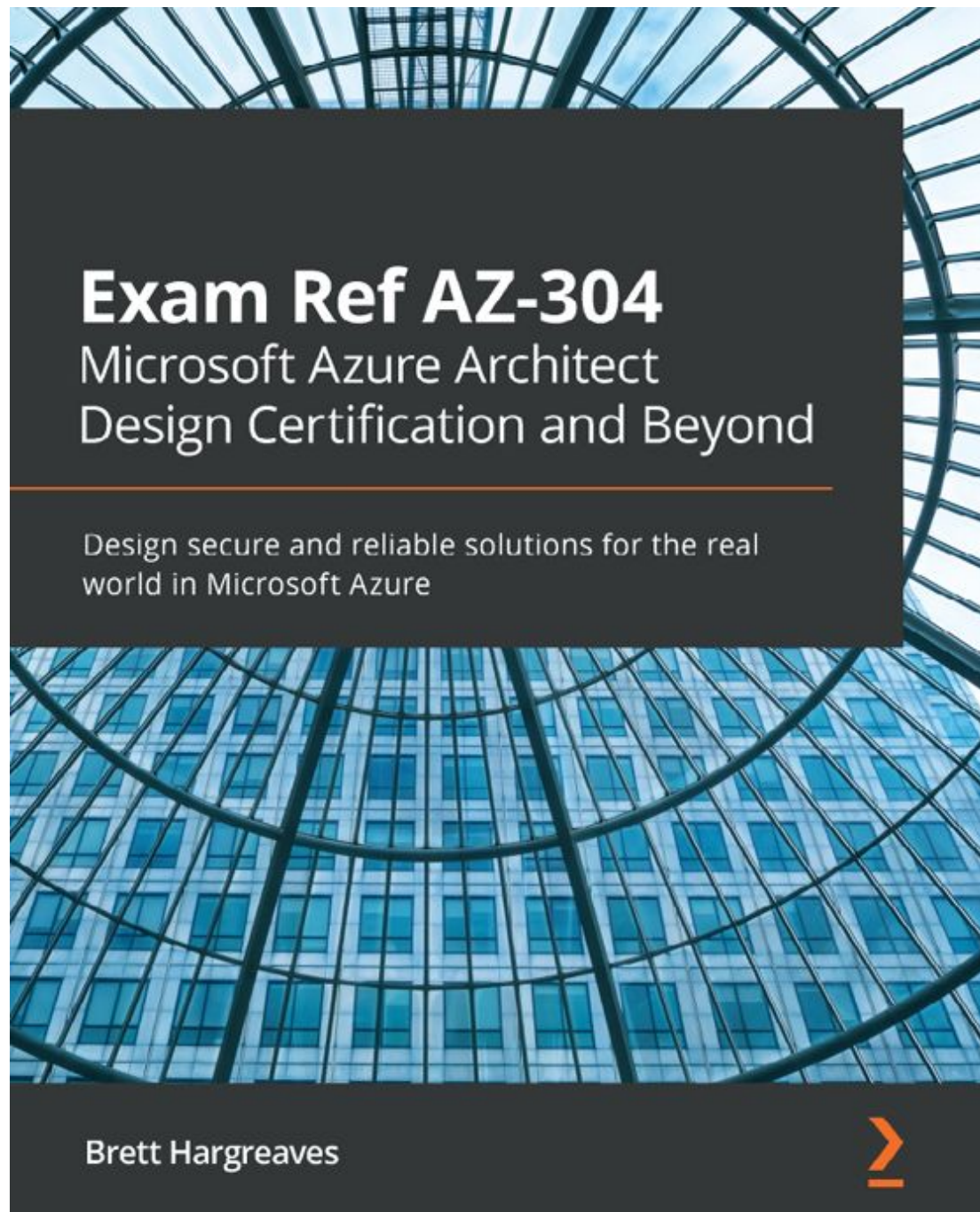
- Spend less time learning and more time coding with practical eBooks and Videos from over 4,000 industry professionals
- Improve your learning with Skill Plans built especially for you
- Get a free eBook or video every month
- Fully searchable for easy access to vital information
- Copy and paste, print, and bookmark content

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at packt.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at customercare@packtpub.com for more details.

At www.packt.com, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on Packt books and eBooks.

Other Books You May Enjoy

If you enjoyed this book, you may be interested in these other books by Packt:



Exam Ref AZ-304 Microsoft Azure Architect Design Certification and Beyond

Brett Hargreaves

ISBN: 978-1-80056-693-4

- Understand the role of architecture in the cloud
- Ensure security through identity, authorization, and governance
- Find out how to use infrastructure components such as compute, containerization, networking, and storage accounts
- Design scalable applications and databases using web apps, functions, messaging, SQL, and Cosmos DB
- Maintain operational health through monitoring, alerting, and backups
- Discover how to create repeatable and reliable automated deployments
- Understand customer requirements and respond to their changing needs

Microsoft Azure Security Technologies Certification and Beyond

Gain practical skills to secure your Azure environment and pass the AZ-500 exam

David Okeyode



Microsoft Azure Security Technologies Certification and Beyond

David Okeyode

ISBN: 978-1-80056-265-3

- Manage users, groups, service principals, and roles effectively in Azure AD
- Explore Azure AD identity security and governance capabilities
- Understand how platform perimeter protection secures Azure workloads
- Implement network security best practices for IaaS and PaaS
- Discover various options to protect against DDoS attacks
- Secure hosts and containers against evolving security threats
- Configure platform governance with cloud-native tools
- Monitor security operations with Azure Security Center and Azure Sentinel

Packt is searching for authors like you

If you're interested in becoming an author for Packt, please visit authors.packtpub.com and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Share Your Thoughts

Now you've finished *Microsoft Azure Fundamentals Certification and Beyond*, we'd love to hear your thoughts! If you purchased the book from Amazon, please [click here to go straight to the Amazon review page](#) for this book and share your feedback or leave a review on the site that you purchased it from.

Your review is important to us and the tech community and will help us make sure we're delivering excellent quality content.