1

Introduction to AWS Accounts and Global Infrastructure

Cloud computing is the on-demand delivery of IT services, enabling customers to provision IT resources such as infrastructure, platform, and software components to host and manage their applications. Cloud computing offers several advantages over traditional on-premises solutions, such as access to cutting-edge technologies, elasticity and scalability, and reduced management overhead. One of the biggest advantages of moving to the cloud is the ability to trade fixed costs for variable costs – this is because customers need not purchase any infrastructure hardware and instead rent capacity on the provider's underlying infrastructure using a pay-as-go model.

Amazon Web Services (**AWS**) is the largest provider of cloud computing services today, followed by Microsoft and Google as per the Gartner 2023 Magic Quadrant for Strategic Cloud Platform Services. AWS offers over 200 services designed to help businesses of all sizes build, design, and deploy scalable, secure, and cost-effective solutions in the cloud. Furthermore, AWS offers access to its global data centers, enabling businesses to deploy their applications closer to their customers without the need to procure and provision infrastructure in those locations. Even small start-ups can access a global customer base and fulfill compliance and regulatory requirements.

This book will teach you all the skills necessary to pass the AWS Certified Developer Associate certification and excel at typical job roles such as cloud developer, junior DevOps engineer, cloud engineer, and many more. It covers AWS architecture design patterns following industry standard best practices and helps you build proficiency in developing, deploying, and debugging cloud-based applications on AWS. This book has also been designed to help you develop real-world practical skills from the ground up. This is achieved by incorporating a scenario throughout the chapters such that, as you learn new technologies and concepts, you also develop the skills to put that knowledge to practical use using the project exercises provided.

We will start by introducing the **AWS Global Infrastructure**, which represents the vast collection of data center facilities located across multiple countries throughout the world, which you can access to design, build, and deploy cloud resources. You will also learn about the various tools used to help effectively manage your AWS accounts. Furthermore, you will also complete a series of exercises that will help you to apply the principles related to having a multi-account AWS architecture in real-world use cases.

Making the Most Out of This Book – Your Certification and Beyond

This book and its accompanying online resources are designed to be a complete preparation tool for your **DVA-C02 Exam**.

The book is written in a way that you can apply everything you've learned here even after your certification. The online practice resources that come with this book (*Figure 1.1*) are designed to improve your test-taking skills. They are loaded with timed mock exams, interactive flashcards, and exam tips to help you work on your exam readiness from now till your test day.

BEFORE YOU PROCEED

*To learn how to access these resources, head over to **Chapter 17**, Accessing the Online Practice Resources, at the end of the book.*

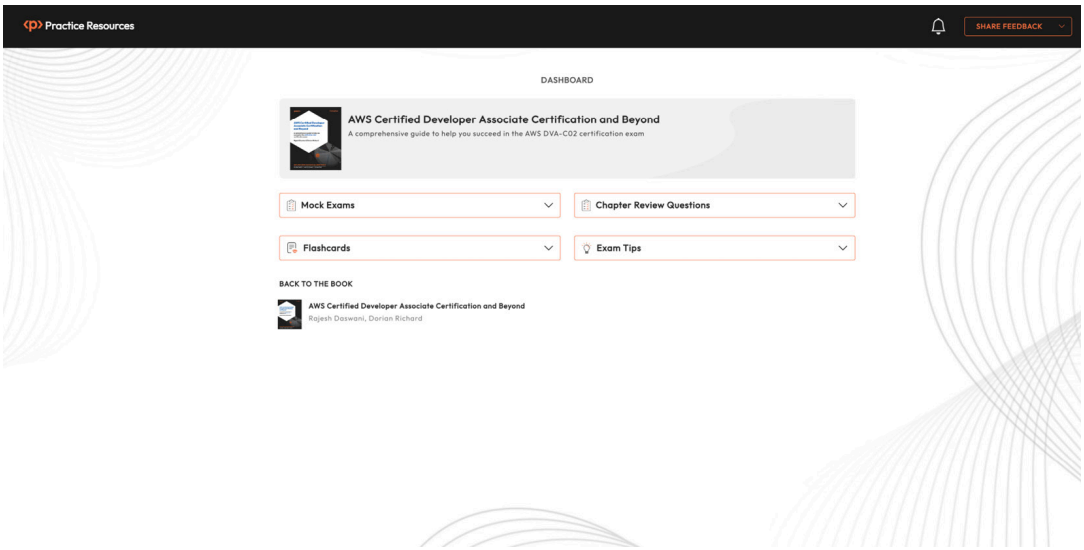


Figure 1.1 – Dashboard interface of the online practice resources

Here are some tips on how to make the most out of this book so that you can clear your certification and retain your knowledge beyond your exam:

1. Read each section thoroughly.
2. **Make ample notes**: You can use your favorite online note-taking tool or use a physical notebook. The free online resources also give you access to an online version of this book. Click the **BACK TO THE BOOK** link from the Dashboard to access the book in **Packt Reader**. You can highlight specific sections of the book there.
3. **Chapter Review Questions**: At the end of this chapter, you'll find a link to review questions for this chapter. These are designed to test your knowledge of the chapter. Aim to score at least **75%** before moving on to the next chapter. You'll find detailed instructions on how to make the most of these questions at the end of this chapter in the *Exam Readiness Drill - Chapter Review Questions* section. That way, you're improving your exam-taking skills after each chapter, rather than at the end.

4. **Flashcards**: After you've gone through the book and scored **75%** more in each of the chapter review questions, start reviewing the online flashcards. They will help you memorize key concepts.
5. **Mock Exams**: Solve the mock exams that come with the book till your exam day. If you get some answers wrong, go back to the book and revisit the concepts you're weak in.
6. **Exam Tips**: Review these from time to time to improve your exam readiness even further.

This chapter covers the following topics:

- Introducing a client scenario for this study
- Introduction to cloud computing and the AWS Global Infrastructure
- Overview of an AWS account
- When one AWS account is not enough
- Accessing the AWS account using the web interface, CLI, and SDKs
- Project tasks – Building a multi-account strategy for TodoPlus Limited

Introducing a Client Scenario for this Study Guide

To help you maximize your learning experience and develop real-world skills, the primary requirement for most technical job roles, this study guide has been designed to teach you how to fulfill a set of requirements for a fictitious company called **TodoPlus Limited**. This study guide will teach you how to architect, build, deploy, debug, and manage cloud-based applications. You will learn application security concepts, best practices, and development and deployment strategies, all of which will be tested in the AWS Certified Developer Associate exam.

This study guide will help you understand how the cloud can address TodoPlus Limited's needs and challenges. By delving into security, scalability, high availability, and effective financial operations (FinOps), you will learn how the cloud can provide solutions that meet these needs and offer the potential for growth and innovation. By the end of this journey, you should clearly understand how cloud computing can be leveraged to drive business success in a realistic scenario. You will also be well-equipped to pass the AWS Certified Developer Associate exam. This exam will test your knowledge and ability to develop applications on AWS, learn how to secure application code, and understand the fundamentals of DevOps by applying continuous integration and continuous delivery as your application design workflows.

Setting the Scene – Background Information

TodoPlus Limited, based in New Jersey, specializes in creating custom productivity apps for small to medium-sized businesses. Their unique approach involves tailoring applications to meet the specific needs of each client. Now, they have a new goal – to develop and sell **off-the-shelf (OTS)** productivity apps directly to the public.

The company, which previously focused on creating apps for clients to use internally or sell to end users, wants to enter the retail market with its own line of products. The first step is launching a "To-Do List" application for individual users by the end of the year. Later, they plan to expand their offerings to businesses.

The board of directors is excited about this new venture and believes it's a market where TodoPlus can make a significant impact.

In this study guide, you will see how the AWS cloud platform can facilitate the application's build, design, hosting, and management for TodoPlus.

Business Needs

TodoPlus aims to create a task management application that's dependable, user-friendly, and accessible to anyone at any time or place. The application must handle millions of concurrent users without compromising service quality. Additionally, given the sensitive nature of the information, security is a significant concern. Lastly, TodoPlus must be able to scale its application offering to support users globally and anticipate uptake from end users, backed by the capabilities of its in-house marketing team, all while keeping operational costs under control.

How the Cloud Meets These Needs

TodoPlus knows that the application will need to be hosted on a cloud platform that can offer the required levels of security, scalability, and fault tolerance and be cost-effective. Let us look at how the AWS cloud can fulfill these core underlying business requirements:

- **Security**: Cloud providers offer a range of built-in security features, including data encryption, two-factor authentication, firewalls, and more. These features enable TodoPlus to protect its users' sensitive data.
- **Scalability**: The cloud provides elasticity, allowing TodoPlus to adapt to demand. If the number of application users increases, TodoPlus can scale up server capacity to meet this demand. Conversely, during periods of low demand, they can scale down resources to minimize costs.
- **High Availability**: Cloud computing services offer high availability by replicating data and applications across multiple data centers in different regions. This means that even if one data center fails, the TodoPlus application will still be accessible to its users.
- **FinOps**: With the cloud's pay-as-you-go pricing model, TodoPlus only pays for the resources it uses. This allows the company to control its costs effectively. Moreover, many cloud providers offer cost

analysis tools to help TodoPlus monitor and optimize their cloud resource expenses.

In summary, the cloud enables TodoPlus to create a task management application that meets its business needs regarding security, scalability, high availability, and effective financial management.

As we move to the next section, you will delve deeper into the fundamentals of cloud computing and explore the extensive capabilities of the AWS Global Infrastructure, providing a solid foundation to understand how it can be harnessed for applications such as our TodoPlus use case.

Introduction to Cloud Computing and the AWS Global Infrastructure

As discussed previously, computing refers to on-demand access and delivery of IT services, which customers can consume over the standard public internet or some form of wide-area network. These services will include compute, network, storage, databases, and **Software as a Service** (**SaaS**) products. Cloud computing has enabled businesses to design and deploy applications without requiring expensive hardware upfront. Instead, they lease/rent required IT infrastructure from such third-party providers.

Of the various providers of cloud computing services, AWS is the largest provider, offering a variety of cloud IT services. These services fall into various categories: **Infrastructure as a Service** (**IaaS**), **Platform as a Service** (**PaaS**), and **Software as a Service** (**SaaS**) solutions. Businesses can consume those services rather than create their own dedicated environments to host applications. With established infrastructure across multiple countries and massive economies of scale,

AWS can offer infrastructure and software service at a fraction of the cost, with redundancy, scalability, high availability, and security.

In the following section, you will learn about the AWS Global Infrastructure, which enables customers across the globe to access AWS services. Furthermore, customers in one location can consume cloud services worldwide, enabling access to a global customer base, and fulfilling any compliance or regulatory needs.

AWS Global Infrastructure

The **AWS Global Infrastructure** is a vast collection of data center facilities across multiple countries globally. The geographical locations where AWS hosts its data center services comprising compute, storage, and network, along with its vast array of cloud services, are known as **AWS Regions**. You will find the map at **https://aws.amazon.com/about-aws/global-infrastructure/**.

Within each AWS region are small groups of data centers that are logically and physically separated by a distance that falls within 100 **kilometers (km)** (60 miles) of each other. These logically and physically separated groups of data centers form what we call **Availability Zones (AZs)**. Furthermore, AWS designs its regions with multiple AZs per region. Most AWS regions have at least three AZs, and some have even more. For example, the North Virginia region (us-east-1) has six AZs.

In the next section, you'll look at the core components of the AWS Global Infrastructure in more detail.

Regions

As explained earlier, AWS regions are physical locations across the globe where AWS hosts its infrastructure facilities. These comprise data centers designed to enable customers to access a vast collection of

infrastructure services with which they can deploy cloud resources, such as compute, network, storage, and database services. Customers can connect to a given region anywhere across the Global Infrastructure.

Choosing a region to provision cloud resources ultimately depends on the use case of the business. Often, this will be based on multiple factors, including the following:

- The requirement to host infrastructure resources closer to your end users, where you can host your applications with reduced network latency.
- The requirement to host infrastructure within political and national borders to adhere to strict data sovereignty and compliance regulations.
- The requirement to isolate groups of resources from each other to facilitate disaster recovery and business continuity use cases.

NOTE

In the case of our fictitious client, TodoPlus, the initial offering for the application will be based in the US to fulfill compliance and regulatory requirements for storing data within the US borders. Should the product be successful, TodoPlus would be looking to expand into Europe and Asia Pacific once all necessary laws and regulatory requirements have been analyzed, and measures are taken to adhere to them.

AZs

Within each AWS region, you will find multiple AZs, which are metropolitan areas housing one or more data center facilities in each region. Each AZ will host hardware components such as servers, storage, and network equipment, all fitted with redundant power, connectivity, cooling, and security controls.

The primary purpose of having multiple AZs in each region is to enable customers to host their applications and workloads in a manner that offers high availability, fault tolerance, and scalability. With multiple AZs, you can host copies or replica application resources across these AZs, which ultimately means that you can continue to serve your customers even if there is an outage of one AZ in the given region.

This is all possible because, although each AZ operates independently, they are still connected over high-speed, high-bandwidth, low network latency, and fully redundant, dedicated metro fiber connectivity.

NOTE

Concerning our company, TodoPlus, their initial choice of region to host their application will be set to the US-East-1 (North Virginia) region. This region is selected because the company will start promoting its new productivity application in local markets. However, they plan to make the application available to customers across the US and later globally.

Edge Locations and Regional Edge Caches

The AWS Global Infrastructure also comprises edge locations and Regional Edge Caches. **Edge locations** or **points of presence (POPs)** offer massive amounts of storage, high-bandwidth networking equipment, and edge computing services that enable data to be accessed, processed, and analyzed closer to the end customers' physical location.

These edge locations are connected to AWS regions through the AWS backbone network. This comprises fully redundant, multiple 100-**Gigabit Ethernet (GbE)** parallel fiber connections that substantially improve throughput and offer low-latency connectivity. You can review the current list and types of edge locations (POPs) at **https://aws.amazon.com/cloudfront/features/**.

Edge locations are different from standard regions and AZs. You **cannot** connect directly to a given edge location to set up resources. Instead, you consume certain AWS services that use these edge locations' storage, caching, and high network connectivity. One service that uses these edge locations is Amazon CloudFront.

Regional Edge Caches are like edge locations. However, they are strategically placed and have a larger storage capacity to hold cache data longer than individual edge locations. Individual edge locations have a shorter time-to-live than Regional Edge Caches, ensuring that stale data isn't hosted too long. If the same cache data is later accessed, an attempt is made to see whether it is still available at regional edge caches before sending a request to the origin.

Amazon CloudFront is a **content delivery network** (**CDN**) service that enables you to efficiently distribute content to end users in a manner that reduces overall latency. With CloudFront, regularly accessed content is cached in the edge location and in regional edge caches, which offer the lowest latency to end users who attempt to access your content. This means those users do not have to fetch frequently accessed content from the origin if it resides in the cache.

NOTE

In the case of our fictitious company, TodoPlus Limited, the application will be hosted in the N.Virginia (us-east-1) region. CloudFront can be used to cache static content such as images, videos, and user guides to help reduce the latency for end users as they access the application from various parts of the US and, ultimately, from across continents when our client expands its offering globally.

Edge locations can allow customers to upload data to AWS storage services such as Amazon S3 over the AWS backbone network, offering low latency and high-bandwidth throughput using a service known as **S3 Transfer Acceleration** (**S3TA**).

The AWS Global Infrastructure also comprises other infrastructure services, including the following:

- **Local zones** are special zones designed to bring compute, storage, database, and other select AWS services closer to end customers' physical locations. This is particularly useful if you require very low latency access to cloud services. Regarding TodoPlus Limited, this will not be necessary as its customer base is geographically dispersed across the US and potentially across the globe.
- **Wavelength zones** are zones where AWS has deployed infrastructure services such as compute and storage services within 5G network providers to help optimize mobile edge computing applications.
- **Direct Connect locations** are designed to establish high bandwidth network connections between clients' data center facilities and the AWS cloud. TodoPlus Limited may wish to set up a Direct Connection to the AWS cloud. This will improve data transfer speeds between on-premises applications and the cloud due to the higher bandwidth capability.
- **Outposts** enable true hybrid cloud computing design by extending AWS infrastructure services, APIs, and tools to customers' on-premises locations. If TodoPlus Limited plans to continue with a hybrid cloud model, an Outpost configuration will enable them to access AWS services locally, allowing low-latency access to certain applications that cannot be hosted in the cloud. If they have applications that need to follow strict compliance or regulatory requirements, then, again, an Outpost setup will help achieve this as all the data will be held locally on-premises.

This section examined the AWS Global Infrastructure and identified some of its core components. Understanding how the Global Infrastructure is architected will enable you to design applications for high availability, scalability, security, and cost-effectiveness.

In the next section, we will look at how you can access the vast array of AWS services via an AWS account as a customer.

Overview of an AWS Account

AWS, a global public cloud provider, offers a comprehensive suite of infrastructure, platform, and software services. An AWS account is necessary to utilize these services, enabling customers to create a wide range of resources and host their applications on AWS.

Each customer needs to have a secure environment within which to create and manage IT resources; this is where the concept of an AWS account comes in. An AWS account, by its very nature, offers an **isolated resource container** that grants secure access to AWS services and enables customers to configure necessary resources to host their applications.

While setting up an AWS account, you must provide an email address and password. These credentials are used to create the primary owner, who has full control over the account. On AWS, we call this owner the **root user**. Every AWS account created will have its dedicated root user. This root user can perform all account and billing-related activities and close the account if no longer required. Each AWS account also provides a natural billing boundary for costs incurred in that account and is associated with a specified billing method. Overall, an AWS account offers a means to securely administer all your resources and is only accessible to you and any entity you choose to grant access to.

In the following section, you'll discover why many companies choose to set up multiple AWS accounts for handling diverse workloads. You'll also learn why this practice of a multi-account strategy is considered a best practice in the industry.

When One AWS Account Is Not Enough

While hosting all your different applications, workloads, and environments in a single account is possible, this can easily become difficult to manage. Each application requires a different environment to support the various stages of its lifecycle, such as development, testing, and production. Hosting multiple environments for multiple applications in a single account is possible, but this can quickly turn complicated. This complexity arises for various reasons, including the need to ensure separation and isolation of workloads and environments for security reasons or budget and cost allocation.

Furthermore, hosting all your application workloads and environments in a single AWS account increases the risk of significant mishaps, which could wipe out all your applications and their different stages of development.

Having a separate account for development, testing, and production and one (or multiple) separate account(s) for experimental workloads makes sense. As depicted in *Figure 1.2*, any adverse events in the **sandbox** (experimental) account will not affect workloads in the other accounts.

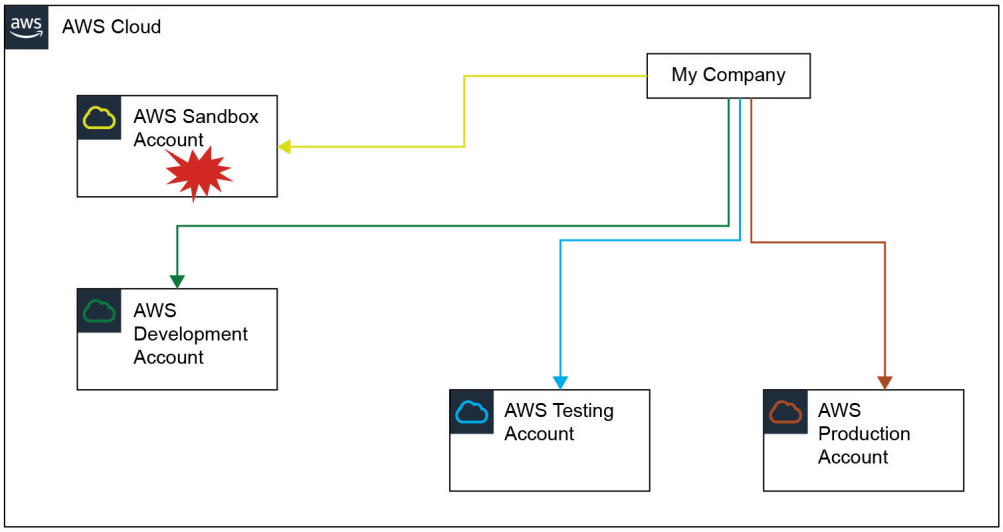


Figure 1.2 – AWS Organization – limiting the blast radius of major disasters

AWS highly recommends adopting a multi-account strategy and provides specific services to help build and manage such multiple accounts. In the following subsection, you'll get to understand how AWS Organizations enable the secure and efficient management of multiple AWS accounts.

Introducing AWS Organizations

You can create several AWS accounts to help you separate different workloads or application life cycles. However, you need a mechanism to secure and manage them. You also need to define policies and permissions in the form of guardrails that allow you to specify what services can be consumed in each account, what resources can be deployed, and even what regions those accounts can deploy applications in. This ensures that you adhere to strict compliance and regulatory requirements for your industry and helps you meet corporate governance and management.

The AWS Organizations service is the ultimate tool to help you effectively provision and manage multiple AWS accounts. To set up AWS Organizations, you must designate an AWS account as the **management account**. This account hosts the actual AWS Organizations service for your business. You can then invite existing accounts or create additional AWS accounts that will become **member accounts** of the organization.

You can also have multiple member accounts in your AWS organization, where some share similar workloads or functions. For example, you may have multiple **development** accounts or multiple **production** accounts. These logical groups of accounts may share similar security requirements as well. To effectively manage these groups, you can create **Organization Units** (**OUs**), logical containers that club

members' accounts within the AWS Organization that share common workloads. So, for example, as depicted in the following diagram, we have three different OUs. Within each OU, you can place the relevant AWS accounts and apply policies and permissions that the members of the OU may share in the organization, as shown in *Figure 1.3*:

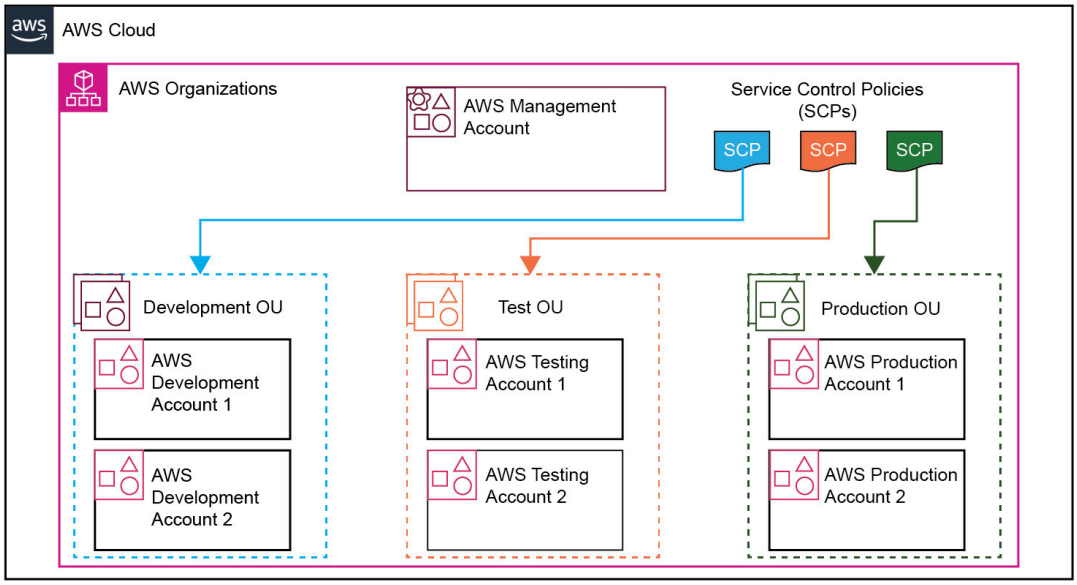


Figure 1.3 – AWS Organizations with OUs and SCPs

The number of AWS accounts required for your business is not a fixed quantity. It varies based on several factors including the functional needs of your business, the complexity of workloads, and the specific security and compliance requirements your business must adhere to. Often, you have some shared services accounts, too – for example, a separate identities account that can be used to centrally manage users (usually representing your colleagues or technical staff) and define the permissions on what they can do in your member accounts. Using a process known as cross-account access or **single sign-on** (**SSO**), discussed in ***Chapter 2***, *Securing Access with AWS Identity and Access Management, Federation, and Amazon Cognito*, you will learn how these users can switch roles into other AWS accounts within your organization and carry out various job functions.

In the next subsection, we look at service control policies, a mechanism to enforce rules in how your AWS accounts are used, what ser-

vices can be consumed, and even which regions you can access.

Service Control Policies (SCPs)

As depicted in *Figure 1.2*, you can apply SCPs that act as **guardrails** on what services can be consumed in each AWS account. SCPs are policy documents written in **JavaScript Object Notation** (**JSON**) format, and they enable you to define what type of resources can be deployed in your AWS account and what actions can be performed against those services. SCPs can also restrict which regions those accounts can provision resources in. The guardrails are designed to help ensure that only approved services can be accessed in your AWS accounts.

NOTE

SCPs do not define what individual users can do in your AWS account. They define the maximum permissions administrators can assign to those users and what actions against AWS services are permitted.

The AWS **Identify and Access Management** (**IAM**) service is then used to assign those permitted permissions to those individual users. We will look at the IAM service in greater detail in the next chapter.

SCPs are disabled when setting up AWS Organizations. When you enable them, a default SCP policy called `FullAWSAccess` is attached to the root and applied to any OUs and AWS accounts in the organization.

NOTE

You can also apply a policy at the account level.

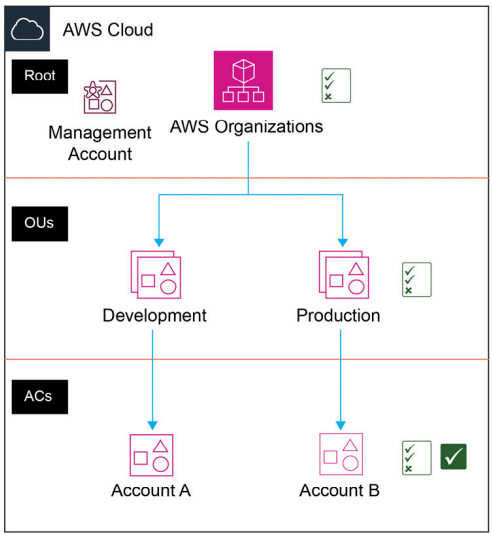
SCPs follow a deny-by-default design. So, unless permission is defined to allow a specific action at all levels from the root, through OUs, and to the accounts in its path, it will be denied. When you first enable SCPs, the `FullAWSAccess` policy is applied at all levels from the root, through OUs, and, finally, directly to accounts; there are no restric-

tions to any services or actions you can perform in your AWS accounts. You should then strategically plan and apply your policies to align with your corporate requirements.

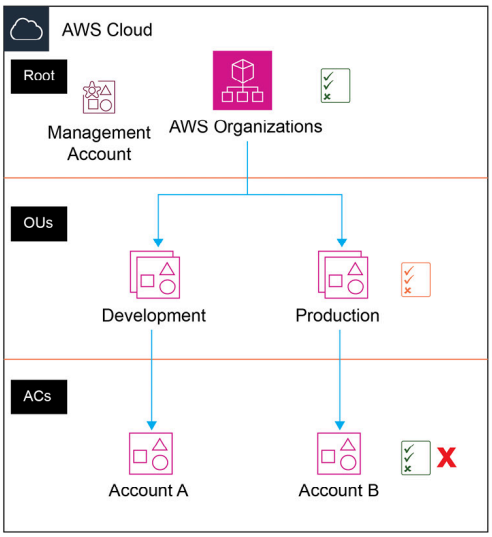
If you detach the **`FullAWSAccess`** policy from a specific OU or account, you must ensure that you replace it with a policy that specifies the services you want to access and the actions you wish to perform. Otherwise, you cannot perform any actions in the account.

If you apply a deny permission at any level in the path of the account – that is, if you apply it at the root or the OU that contains the account – then the deny rule will override any conflicting allow permission.

In the following diagram, *Figure 1.4*, you will note two different configurations. In configuration A, permission to perform an action is applied at all levels, which results in the permission being allowed in Account B. However, in configuration B, permission to perform an action is being denied at the OU level, specifically applied to the Production OU. This results in the permission being denied in Account B, even though there is an explicit Allow rule being applied to this account. This is because a deny rule will always override an allow rule. Furthermore, allow permissions must be explicitly defined at all levels as the default rule is to deny the action.



Configuration A
SCP permits action at all levels



Configuration B
SCP denies action at OU level

Figure 1.4 – Comparing allow and deny rules for SCPs

SCP policies can have multiple statements with allow and deny permissions. This makes it easy to draft and apply a single policy with both allow and deny permissions to an OU and account.

For example, you can attach the following policy to an OU (or member account). This would result in being able to perform all actions against all services except Amazon S3:

```
{
    "Version": "2012-10-17",
    "Statement": [
     {
            "Sid": "AllowsAllActions",
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        },
        {
            "Sid": "DenyAmazonS3",
            "Effect": "Deny",
            "Action": "S3:*",
            "Resource": "*"
        }
    ]
}
```

It's perfectly fine if the structure of the policy seems a bit complex at this point; you'll delve into the details of JSON policy documents in the upcoming chapter.

With AWS Organizations, you can also manage the costs associated with your AWS accounts more effectively using the **Consolidated Billing** feature, which is included as part of the service. The manage-

ment account is responsible for charges incurred by individual member accounts, and you are provided with a consolidated view of those charges. One of the key benefits of this feature is **volume discounts**. By combining the costs across your accounts, you can get discounts when your total charges cross certain volume discount thresholds.

Furthermore, you can use AWS Organizations to share certain resources across your member accounts. For example, you may wish to share certain network resources, such as VPC subnets, or share reserve instances purchased across member accounts. VPC subnets are further discussed in ***Chapter 4***, *Building Private Networks in the Cloud with Amazon VPC*.

This section analyzed the use case for architecting a multi-account strategy on AWS. We also looked at how to centrally create and manage multiple AWS accounts and apply SCPs to establish guardrails on what services can be consumed in those accounts. By now, it is evident that meticulous preparation is essential before starting your cloud journey with AWS, incorporating security best practices, and enforcing corporate governance.

In the next section, we look at another AWS service that allows you to set up and manage multiple AWS accounts following recommended guidelines and build landing zones.

AWS Control Tower

As discussed, planning and creating a multi-account strategy for your AWS journey will enable you to ensure higher levels of security, governance, compliance, and cost efficiency. However, building a multi-account environment can become very complex and time-consuming. You need to decide how many AWS accounts you will need, whether certain accounts will be used for internal shared services, and how many environments you will need to cater to.

AWS offers tools to help you build multi-account environments that follow best practices and adhere to *well-architected* principles using **AWS Landing Zone**. These blueprints will help design and architect a multi-account deployment, offering IAM, governance, data security, and audit logging capabilities.

To help automate the deployment of multiple accounts using these recommended blueprints, AWS offers a service known as AWS Control Tower. This service can automatically implement your landing zone to include the following:

- Your AWS Organizations and any member accounts for your various workloads and environments
- An IAM service that offers AWS SSO default directory services
- Account federation using SSO
- Centralized logging using AWS CloudTrail and AWS Config

Using AWS Control Tower, you can also select pre-packaged governance rules for security, operations, and compliance and apply them to the entire enterprise or specific groups of accounts.

The AWS Control Tower service also has a dashboard feature enabling you to view your AWS environment. Using the dashboard, you can see the various OUs and accounts provisioned and the number of guardrails defined and applied to the various OUs and accounts.

In this section, we looked at how you can automate the design and implementation of your multi-account strategy using AWS-recommended blueprints that adhere to its well-architecting principles.

In the next section, we will have a look at the options available to access your AWS account(s).

Accessing Your AWS Account Using the Web Management Console, CLI, and SDKs

There are several ways to access your AWS account and the services offered on the platform. These include the AWS web console, **Command-Line Interface** (**CLI**, and **Software Development Kits** (**SDKs**).

Let us have a look at each of these options in detail.

AWS Web Management Console

You can access your AWS account using an intuitive web interface. The URL to sign into the web console is **https://aws.amazon.com/console/**.

Several exercises throughout this book will be completed using the AWS web management console and you will gain hands-on experience in using the console. The following is a screenshot of the AWS **Console Home** page:

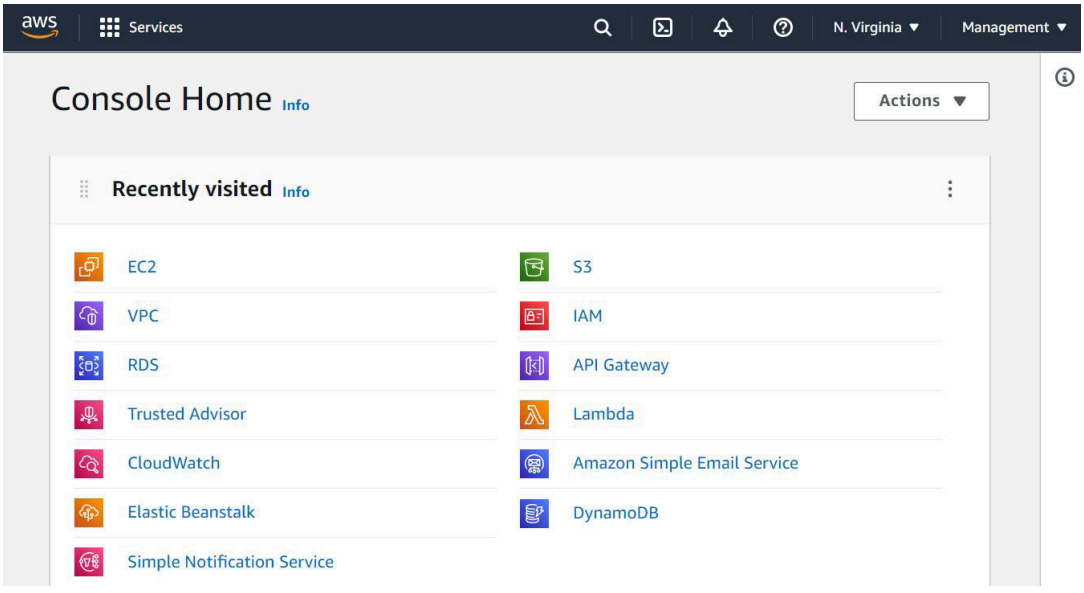


Figure 1.5 – AWS Management Console

To access your AWS account using the web interface, you must either use your root user account or an IAM user account (discussed in the next chapter). The root user owns the AWS account and has full access to all services and resources within the account.

As a best practice, you should not use the root user for day-to-day operations. AWS enables you to create IAM users, which are additional user accounts you can create in your AWS account. These users can represent your colleagues and act on behalf of applications needing access to AWS services. IAM users can be granted access permissions based on their roles, which allows them to interact only with the specific services required for their job responsibilities. This allows you to enforce the **principle of least privileges**, which may involve assigning administrative permissions to key individuals. This way, you can also ensure that the root user credentials are only used when necessary.

Next, you will explore how you can programmatically access your AWS account using the CLI.

AWS CLI

You can programmatically access your AWS account using the CLI. This allows you to issue commands directly to configure services and provision resources from your computer's Command Prompt (if you are using Microsoft Windows) or the Terminal window (for macOS and Linux).

The AWS CLI tool needs to be downloaded and installed on your computer to use; you can access the tool at **https://aws.amazon.com/cli/** for your specific operating system.

The best thing about the CLI is that you will often find it easier and quicker to get things done rather than issue multiple mouse clicks for the same set of tasks. Furthermore, with the CLI, you can still access all public **Application Programming Interface** (**APIs**) of AWS services, and you can further develop scripts to issue multiple commands to perform multiple tasks. These can be both Bash scripts and Windows PowerShell scripts.

To use the CLI, you still need to authenticate against your AWS account with a set of credentials. Unlike a username and password required for web console access, you must configure a set of access keys for use with the CLI. Each IAM user can have up to two access keys. These keys are like usernames and passwords in that they comprise an **access key ID** (like a username) and a **secret access key** (like a password).

Interestingly, you do not choose your own access keys. AWS generates these keys for you, which are specific to an IAM user of a specific AWS account. This means you do not need to specify the account ID of the AWS account you are trying to log in to, which you need to provide when using the web console.

Next, AWS also offers an online version of the CLI via its new **AWS CloudShell** service.

AWS CloudShell

Rather than use the command-line tools on your computer, you can use the AWS CloudShell service instead. This is a browser-based shell for interacting with AWS services and resources securely.

You access the CloudShell service via the AWS web console. This is located as a link icon in the top-right corner of the console screen, as per the screenshot shown here:

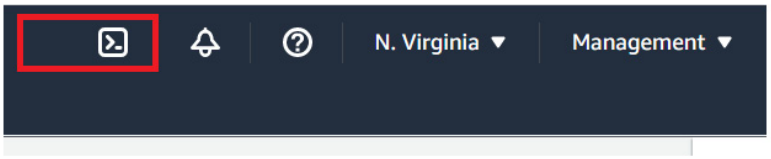


Figure 1.6 – AWS CloudShell service

When you access the AWS CloudShell service, it uses the credentials that were used to log in to the AWS web console. So, there is no need to preconfigure any access keys as you would for AWS CLI.

Next, we will look at AWS SDKs. These are sets of tools, libraries, and documentation that enable developers to build applications that integrate with the AWS ecosystem.

AWS SDKs

An SDK, otherwise known as a devkit, is a collection of tools and programs that allows developers to create an application for a specific platform. SDKs will include libraries, documentation, code samples, and processes that the developer embeds in their application to programmatically interface with a platform – in this case, AWS.

In AWS, you get different SDKs for different programming languages. So, for example, you can get an SDK for Python called Boto3, which provides Python APIs for AWS infrastructure services. This enables you to build Python applications that can interact with services such as Amazon S3, Amazon EC2, Amazon DynamoDB, and others.

This section examined various options to access your AWS account, configure services, and provision resources interactively and programmatically. In the next section, you will do a series of exercises to help you set up your AWS account and define a simple multi-account architecture.

Technical Requirements

To complete all the practical exercises in this chapter, you must create three separate AWS accounts: a **management** account, a **development** account, and a **production** account. Most of the exercises must be performed in the development account. The technical details of having these accounts were explained in the section titled *When One AWS Account Is Not Enough*. Depending on your business use case, you will likely have many more accounts, but to successfully complete the exercises in this chapter and illustrate how these accounts are created and centrally managed, you will need only three AWS accounts.

To set up your initial AWS account, you must have access to a mobile phone and a credit card. Although most of the exercises in this fall within the **Free Tier** offering on AWS, specific labs may cost a small dollar amount. We recommend setting a budget of 25 USD to complete all the labs and exercises in this study guide. We will demonstrate how to set up a billing alert in your AWS account later in this chapter in *Project Task 1.8 – Configuring Billing Alerts*.

You also need a unique email address for each AWS account you create. This email address (and the password you choose when you set up the AWS account) is known as the **root user** of the account and is the main owner of the account. The following subsection explains how you can use one Gmail account (mailbox) with three email aliases to help you effectively have three email addresses for your three AWS accounts. This makes the management of your emails easier as well since, ultimately, you only have one Gmail mailbox.

A Gmail Account

Gmail allows you to use task-specific email addresses pointing to one mailbox account. This is very useful if you want to avoid having more than one email mailbox to manage but still need multiple email addresses. With Gmail, you create one email mailbox, such as **`your-email-id@gmail.com`**, where **`your-email-id`** is a string of characters and/or numbers of your choice. Once you have created this email ac-

count, you can append the email ID with the plus sign (**+**) followed by another term to depict the purpose of that email address. For example, we recommend using task-specific email addresses as follows:

- **`your-email- id+management@gmail.com`** for the AWS management account
- **`Your-email- id+development@gmail.com`** for the AWS development account
- **`Your-email- id+production@gmail.com`** for the AWS production account

Each of these email addresses points to the same mailbox – in this case, **`your-email-id@gmail.com`**. When you set up the individual AWS accounts, a verification email for each account will be sent to the same Gmail mailbox, and you can access all three verification requests from this single Gmail mailbox account.

Project Tasks – Building a Multi-Account Strategy for TodoPlus Limited

While understanding theoretical knowledge of the various services offered on the AWS platform is paramount to passing the exam, developing real-world, hands-on experience is critical if you are to succeed as a developer or even a DevOps engineer on the AWS platform. This is what companies looking to hire new talent are always on the lookout for.

As discussed earlier in this chapter, you will apply the knowledge gained in the study guide to fulfill the business requirements of a fictitious company called **TodoPlus Limited**. This will help you develop much-needed hands-on experience to help you secure your next job role. In the following project task series, you will design and build a

multi-account environment for TodoPlus Limited. You will also set up an AWS Organization and define the necessary OUs and SCPs.

Project Task 1.1 – Setting up Your First AWS Account

In this exercise, you will set up your first AWS account for our client, TodoPlus Limited. You will be using this AWS account to configure an AWS Organization from which you can create the additional two accounts discussed in the *Technical Requirements* section of this chapter. You will use these accounts throughout this exam guide to gain valuable hands-on experience and prepare for the *AWS Certified Developer Associate* exam:

1. Using your web browser, navigate to **https://aws.amazon.com/free/**.
2. Click the **Create a Free Account** button to set up your first AWS Free Tier account. You will be presented with a signup screen.
3. For **Root user email address**, provide your **management account email address** as previously discussed in the *Technical Requirements* section.
4. For the account name, enter the value **`Todo Plus Management`**.
5. Click on the **Verify email address** button, as shown in *Figure 1.7*:

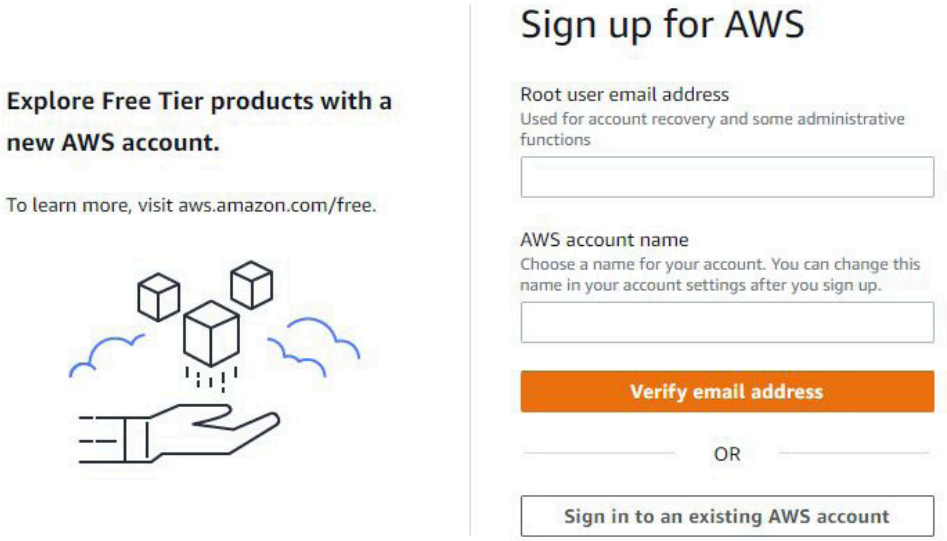


Figure 1.7 – Setting up an AWS Free Tier account

6. You will then be presented with a **Confirm who you are** dialog box, where you must verify the email address you used to create this account. Log in to your Gmail account and obtain the verification code sent to you. Type the code in the **Verification code** textbox and click the **Verify** button, as per *Figure 1.8*:

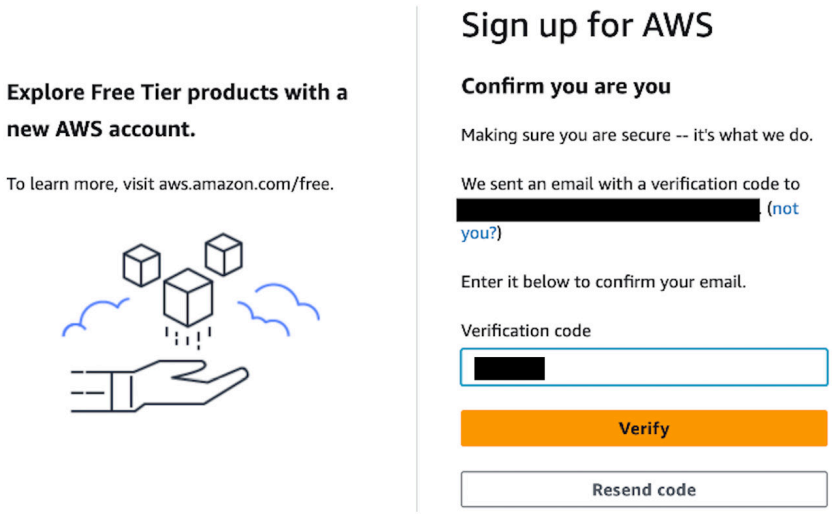


Figure 1.8 – Providing the verification code

7. You need to create a password for this account. This will be the root user's password. Provide a complex password comprising lower-

case letters, uppercase letters, numbers, and at least one non-alphanumeric character. Click **Continue (step 1 of 5)**.

8. For **Contact information**, provide your full contact and address details. Select the **Personal - for your own projects** option. Ensure that you select the checkbox stating that you agree to the terms of the AWS Customer Agreement and click **Continue (step 2 of 5)**.
9. Provide your **Billing Information** details and click **Continue (step 3 of 5)**. You will need access to a credit or debit card and your billing address details.
10. The next step involves providing your mobile phone number for verification purposes. You can choose to have a text message (SMS) sent to you or a voice call. To complete, click on **Text Message (SMS)** and provide your phone number.
11. Type in the characters in the captcha box and click on **Send SMS (step 4 of 5)**.
12. You will now be presented with a **Confirm your identity** page. Check your phone for the code from Amazon and then type the code in the **Verify code** textbox on your signup page, as shown in *Figure 1.9*:

Figure 1.9 – Signing up for AWS – Verify code page

13. As a next step, you will need to select an AWS support plan for your account. For now, choose the **Basic support – Free** plan and click the **Complete sign up** button. You will now see a congratulations screen and will be able to navigate to the main sign-in page.

Congratulations, you have now created your first AWS account! You can now log in to your AWS account using the email address and password you defined earlier. Next, you should consider securing your account further. Right now, it is possible to access your account with just an email address and a password. AWS offers another mechanism to strengthen your account security: multi-factor authentication.

Multi-factor authentication (**MFA**) provides multiple authentication mechanisms before granting access to the service or resource. On AWS, this takes the form of what you know and what you have. The what you know part refers to the email address and password you have memorized. It's a set of credentials that you know. The what you have part further enhances your account security because you need a dynamic piece of information that changes regularly when logging in. If you have an online bank account, you have probably used MFA before: you log in to your account with a username and password and then also receive a PIN to your mobile device, which usually consists of a time-sensitive code that you need to provide as part of your authentication process.

Similarly, you can set up your account on AWS with MFA. In addition to an email address (or a username for IAM users discussed in the next chapter), you can also set up an MFA device. This can be a physical or virtual device, such as an MFA app on your smartphone. The MFA device generates a **one-time PIN** (**OTP**). This OTP is time-sensitive, and you must supply the PIN quickly during your authentication process.

In the next project task, you will learn how to set up MFA for the root user of your management account. You will need access to a smartphone with internet access to complete the next project task.

Project Task 1.2 – Configuring MFA for Your Root User

In this exercise, you will set up and configure MFA for the root user of your AWS management account:

1. Before setting up MFA on your management account, you must install an MFA application on your smartphone. Google Authenticator is one such app that you can download and install on your phone. You will find it on both Apple App Store and Google Play Store.
2. Log in to your AWS account.
3. From the top search bar, search for **`IAM`** and navigate to the IAM web console.
4. On the main **IAM** dashboard, you should see a security recommendation suggesting that you add MFA for the root user, as per the following screenshot:

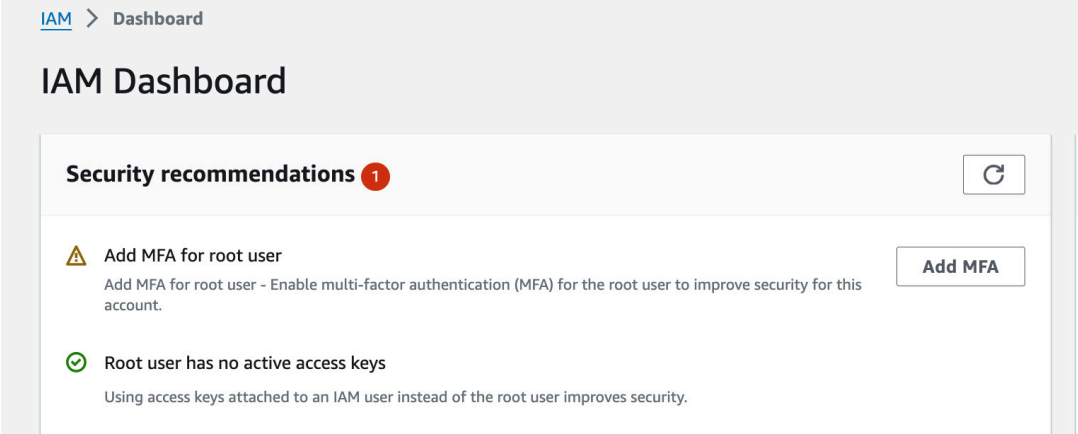


Figure 1.10 – Setting up MFA for the root user

5. Click the **Add MFA** button.
6. Provide a name for your MFA device.
7. Select the **Authenticator app** option under **MFA device**.
8. Click **Next** and you will be directed to the **Set up device** web page.
9. You will need to perform the following steps on your smartphone:
    1. Launch the Google Authenticator app on your phone. If prompted, ensure that you allow access to the camera.
    2. Click the plus (+) icon in the Google Authenticator app and select the **Scan a QR code** option. This will launch your camera app on the phone. There is also the option to enter a setup key if your camera is not working.

3. Back in the AWS Management Console, within the wizard, click the **Show QR code** link on the **Set up device** page, as per the following screenshot:

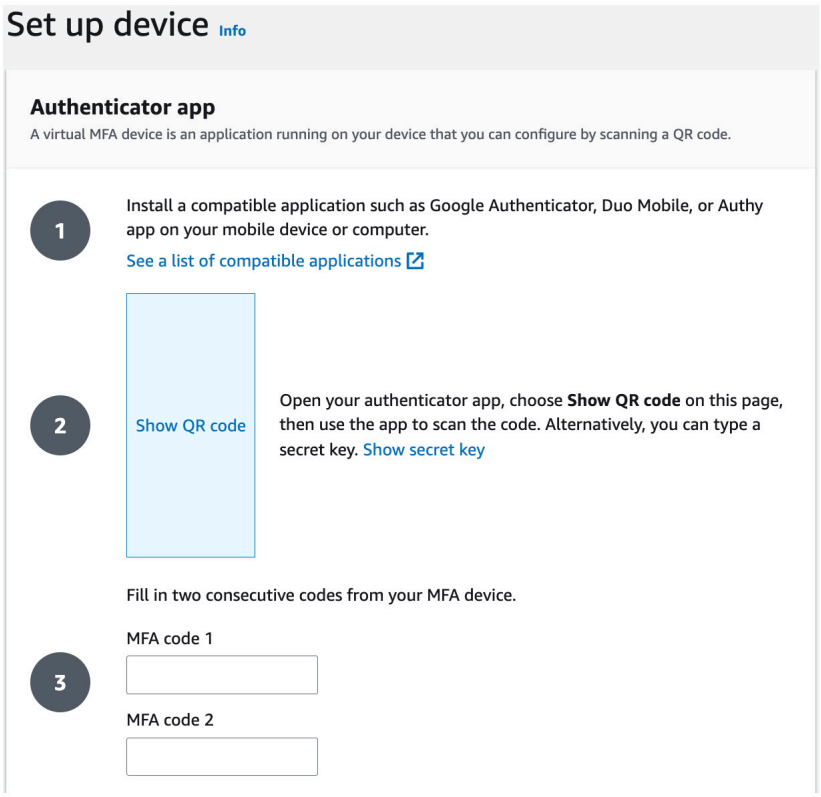


Figure 1.11 – Show QR code

4. Your QR code will be displayed. Position your smartphone camera as though you want to take a picture of the QR code.
5. The app will scan the QR code and add the AWS account to the device. It will display an OTP comprising six digits that expires after a few seconds, after which, another PIN is displayed, and this cycle of new PIN generation continues after a few seconds. Provide the PIN in the sequence displayed for **MFA code 1** and **MFA code 2**, and ensure you click the **Add MFA** button before the second PIN expires. This will associate the Google Authenticator app and the specific virtual device on your phone with your AWS account.
6. When you next log in to your AWS account and provide your root user credentials, you will be prompted to provide the OTP. You will need to launch the app on your phone and then provide the current displayed OTP for your AWS account to be able to log in.

In the next task, you will learn how to configure your AWS management account with AWS Organizations so that you can centrally create and manage other AWS accounts.

Project Task 1.3 – Setting up an AWS Organization

In this exercise, you will log in to the AWS account you created and configure the **AWS Organizations** service. For our client, TodoPlus, this AWS account will be the management account for your AWS Organizations configuration, allowing us to create additional AWS accounts for the development and production environments for our application:

1. Log in to your AWS account to access the **Console Home** page. From this page, you can access all AWS services by clicking on the **Services** link in the top-right corner of the screen or directly searching for the service name in the search box at the top.

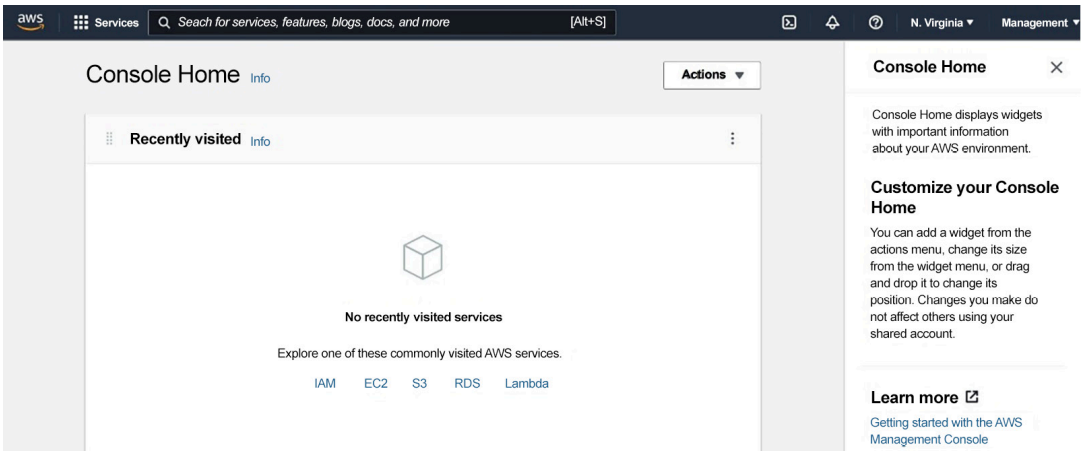


Figure 1.12 – AWS Management Console

2. In the top search box, type in **`AWS Organizations`** and select the service from the drop-down menu, as per the following screenshot:

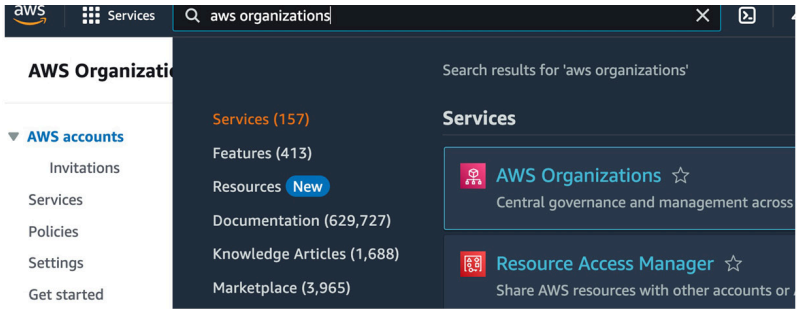


Figure 1.13 – Searching for services using the search box on the AWS Management Console

3. You will be directed to the **AWS Organizations** console page.
4. Click the **Create an organization** button in the top-right corner of the screen. This will create your AWS organization and configure your AWS account to become the primary root management account, as shown in *Figure 1.14*. This account can then host other member AWS accounts, apply security policies, and act as a central billing account for all members' AWS accounts.

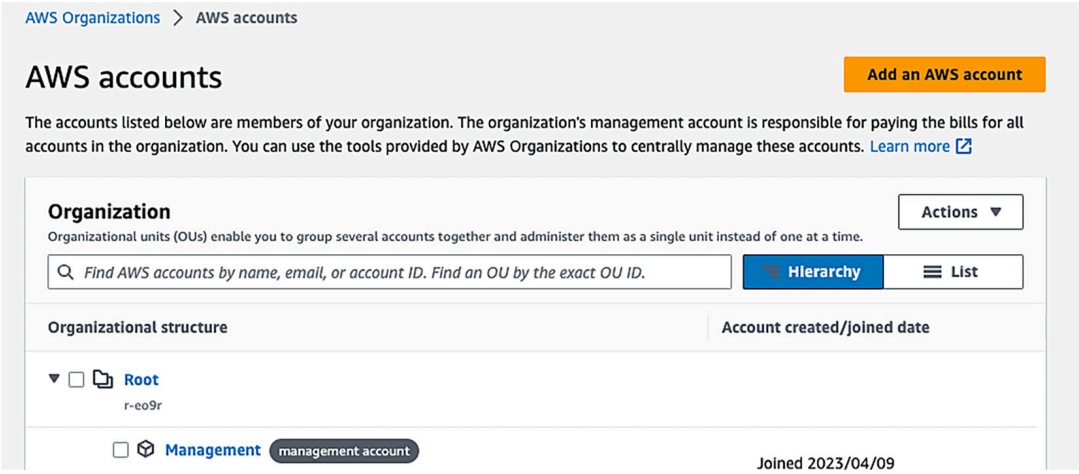


Figure 1.14 – AWS Organizations

Now that you have created your AWS Organizations, it's time to configure some OUs. These OUs enable us to manage multiple accounts that share common workload types, for example, software development environments such as development, testing, and production.

5. Click the checkbox to select **Root** under **Organizational Structure**.
6. Click on the **Actions** drop-down menu and select **Create new** under **Organization unit**; you will be taken to the **Create organiza-**

tional unit in the Root page.

7. Type **`DevelopmentOU`** under **Organizational unit name**.
8. Click **Add tag** in the **Tags** section. Tags are key-value pairs that you can add to AWS resources to help identify, organize, and secure your AWS resources. You can have up to 50 tags per resource. Create two tags, one with a key of **`Name`** and a value of **`DevelopmentOU`**, and the other with a key of **`Project`** and a value of **`ProductivityApps`**, as per *Figure 1.15*:

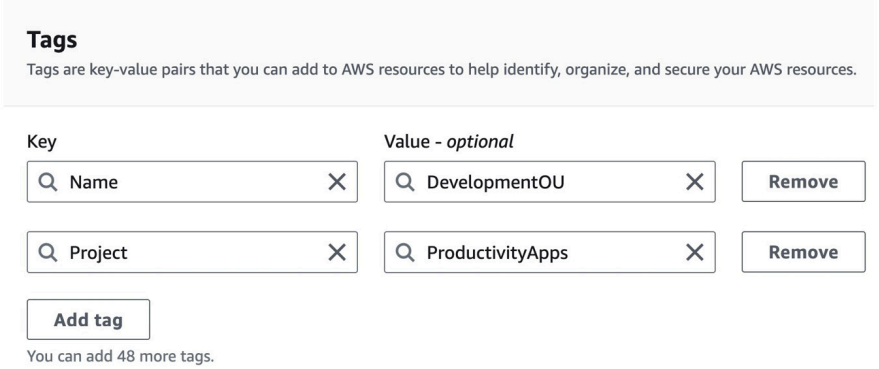


Figure 1.15 – AWS Organization Tags

9. Click **Create organizational unit**.
10. Repeat *steps 5* to *9* to create another OU for **ProductionOU**, so that you now have two OUs in your organization, as per *Figure 1.16*:

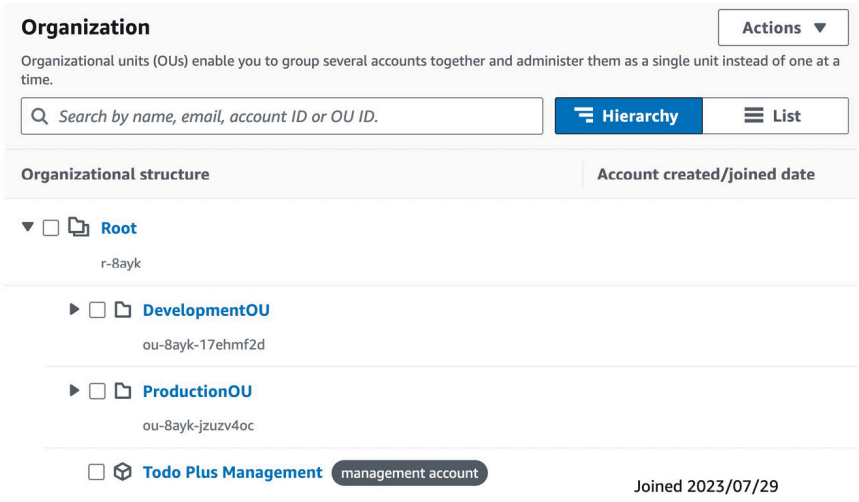


Figure 1.16 – AWS Organizations OUs

Now that you have created your OUs for the organization, it is time to create additional AWS accounts, which is described in the next

exercise.

Project Task 1.4 – Creating the Development and Production AWS Accounts

In this exercise, you will create two additional AWS accounts: development and production. Later in this study guide, you will learn how to build workloads in those accounts and will also be introduced to how application development will transition from the development to production environments following best practices. Follow these steps:

1. Log back into your AWS account and navigate to the **AWS Organizations** management console if necessary.
2. From the left-hand menu, ensure that **AWS Accounts** is selected.
3. In the right-hand pane, click on the **Add an AWS account** button.
4. You will have the option to create a new AWS account or to invite an existing one. Select the **Create an AWS account** option.
5. Under the **Create an AWS account** section, provide a name for your AWS account – in this case, it will be `TodoPlus Development`.
6. For the email address, provide the development email alias, as explained in the *Technical Requirements* section earlier in this chapter.
7. You will also notice that the IAM role is prefilled with the role of **OrganizationAccountAccessRole** (discussed in detail in the next chapter). This enables the management account to access resources in member accounts.
8. Click **Add tag** in the **Tags** section. Create two tags, one with a key of `Name` and a value of `DevAccount`, and the other with a key of `Project` and a value of `ProductivityApps`.
9. Click **Create AWS account**.
10. AWS will take a few moments to create your new AWS development account.

Each AWS account you create in this way will have a root user for that account. The root user is defined by the email address you use to create the account. Note that you did not set a password. To define a password for the root user of your member accounts, you must request a password reset on those accounts from the main AWS sign-in page. A password reset link will be sent to the member AWS account root user's email address, allowing you to set up a complex password. You can then log in to those member AWS accounts.

Repeat *steps 1* to *10* to set up another AWS account for production workloads. Ensure you name this account **`Todo Plus Production`**. For the tags, create one with a key of **`Name`** and a value of **`ProdAccount`**, and the other with a key of **`Project`** and a value of **`ProductivityApps`**.

Now that you have set up your development and production accounts, it would be advisable to place these accounts in the appropriate OUs you had set up in the previous exercise, which you will configure in the next exercise.

Project Task 1.5 – Assigning AWS Accounts to Their Respective OUs

In this exercise, you will be associating the accounts you created in the previous exercise with their relevant OUs:

1. While still in the **AWS Organizations** console, click to select the **Todo Plus Development** account under **Organizational structure**.
2. Click **Move** from the **Actions** drop-down menu under the AWS account.
3. You will be navigated to the **Move AWS account 'Todo Plus Development'** page.
4. Select **DevelopmentOU** under **Destination**.
5. Click the **Move AWS account** button.

6. Repeat *steps 1* to *5* for the **Todo Plus Production** account and ensure that you move the account to **ProductionOU**, as shown in *Figure 1.17*:

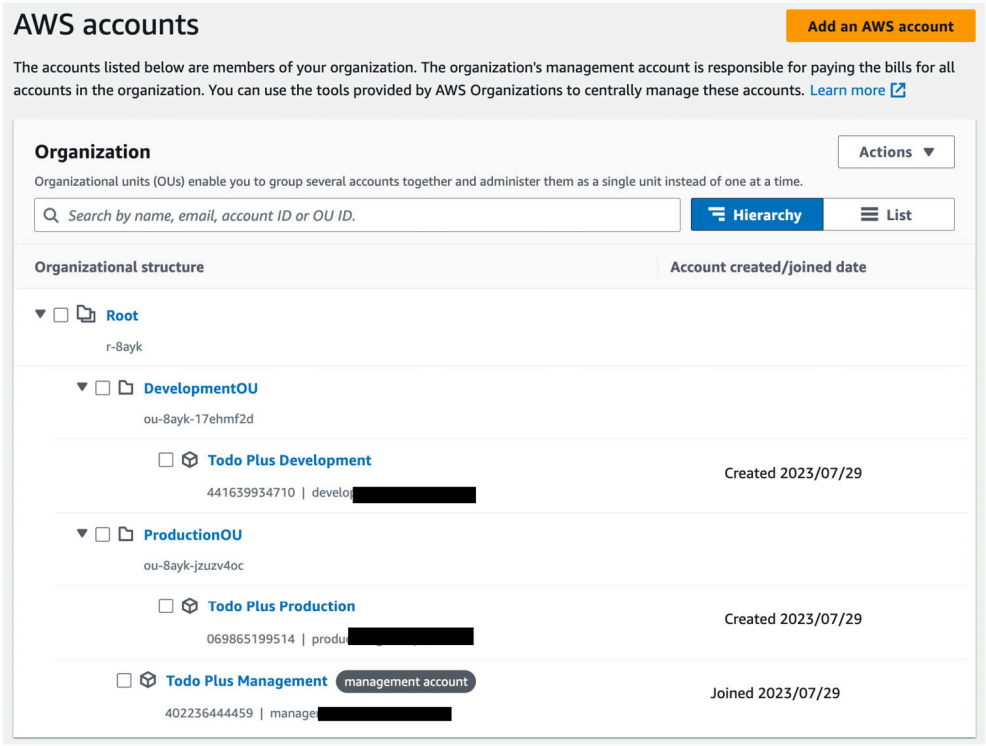


Figure 1.17 – Structuring your AWS accounts with OUs

In this exercise, you have moved the AWS accounts you created earlier into their relevant AWS Organizations OUs. In the next exercise, you will learn how to enable SCPs and apply a new SCP to your OUs.

Project Task 1.6 – Setting up SCPs

In this exercise, you will enable SCP policies and apply an SCP for both the development and production AWS accounts:

1. If necessary, log back into your AWS account and navigate to the **AWS Organizations** console.
2. From the left-hand menu, click on **Policies**.
3. In the right-hand pane, click on **Service Control Policies**, where you will be taken to the SCPs page.
4. Click on the **Enable service control policies** button.

5. Your AWS organization's SCP feature has now been enabled. You will also notice that the default policy applied to the management account, OUs, and member accounts is the **FullAWSAccess** policy. You can now create various SCPs and apply them to both OUs and directly to AWS accounts. These policies can restrict various types of access, services, and even which regions you can access. One SCP that you should consider creating is a policy to deny member accounts from leaving the organization. Recall that every AWS account has a root user. This root user can remove the account from an AWS Organization. In large enterprise environments, you may wish to enforce a policy preventing root users from removing their accounts from the organization, whether due to an accident or malicious intent. In the next steps, you can configure a policy to deny root users of member accounts this ability.
6. While in the **AWS Organizations** console, click on **Policies** from the left-hand menu and then click the **Create policy** button.
7. For **Policy Name**, enter `DenyOrgLeave` and provide an appropriate description for your policy.
8. Scroll further down and in the **Edit statement** block, replace the existing policy with the following policy:

```
{"Version": "2012-10-17",
```

```
    "Statement": [
```

```
        {
```

```
            "Effect": "Deny",
```

```
            "Action": [
```

```
                "organizations:LeaveOrganization"
```

```
            ],
```

```
            "Resource": "*"
```

```
        }
```

```
    ]
```

```
}
```

Recheck your policy as per *Figure 1.18* shown here:

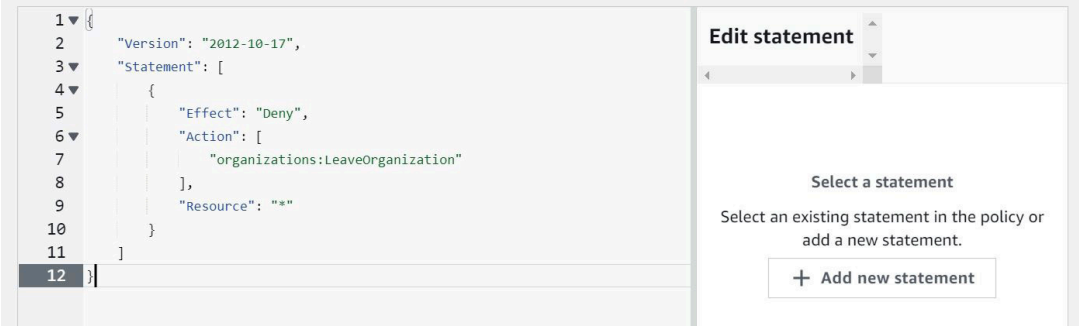


Figure 1.18 – SCP to prevent administrators of member accounts from leaving the organization

9. Scroll down and click the **Create policy** button. You should now be able to see the list of policies available in your organization, as shown in *Figure 1.19*:

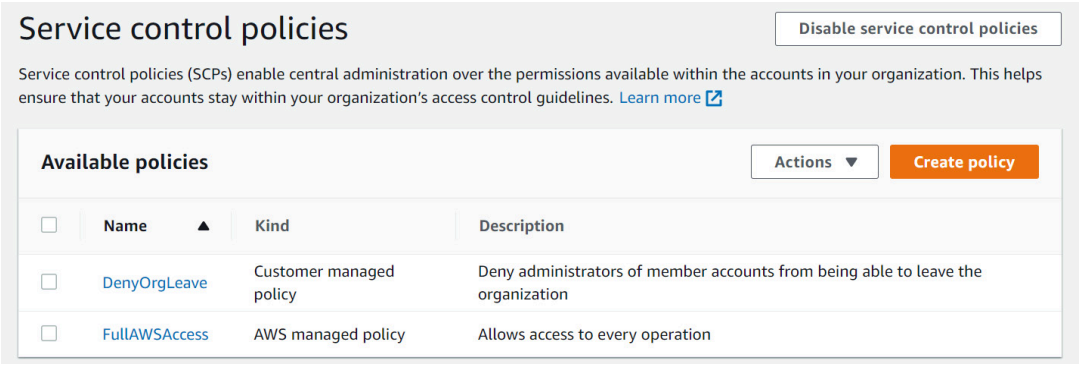


Figure 1.19 – List of available SCPs

10. Click to select the **DenyOrgLeave** policy and then, from the **Actions** drop-down menu, click **Attach policy**.
11. Next, click both the **Production** and **Development** OUs.
12. Click the **Attach policy** button.
13. From the left-hand menu, click **Accounts**.
14. Next, in the right-hand pane, expand the **Development** OU.
15. Click on **Todo Plus Development account**. This will take you to the member account details page.

16. In the lower half of the right-hand pane, click the **Policies** tab. You should find both the **FullAWSAccess** and **DenyOrgLeave** policies listed. Some full access permissions are also inherited from the root account.

In this exercise, you configured SCPs that are not enabled by default and then set up a new SCP to prevent administrators of member accounts from leaving the organization. In the next exercise, you will test out this policy on the **Todo Plus Development** account.

Project Task 1.7 – Testing SCPs

To complete this exercise, log in to the **Todo Plus Development** AWS account. Since you only know the email address for the account but not the password, you will need to reset the password. You must first provide your email address and complete the Captcha challenge. Next, follow the processes on the root user sign-in page to reset the password by clicking on the **Forgot Password** link:

1. You will be sent a password reset email to the address you provided when setting up the **Todo Plus Development** AWS account earlier. Ensure that you use the link in the email to reset the password. Once you have reset your password, you can log in to the AWS account.
2. Search for **`AWS Organizations`** and navigate to its Management Console page. You will note that this account is a member of the **Management** AWS organization.
3. Scroll further down and click on the **Leave this organization** button. You will be prompted to confirm your action. Click the **Leave organization** button.
4. You should see an error message that reads **You don't have permissions to access this resource.**, as shown in *Figure 1.20*:

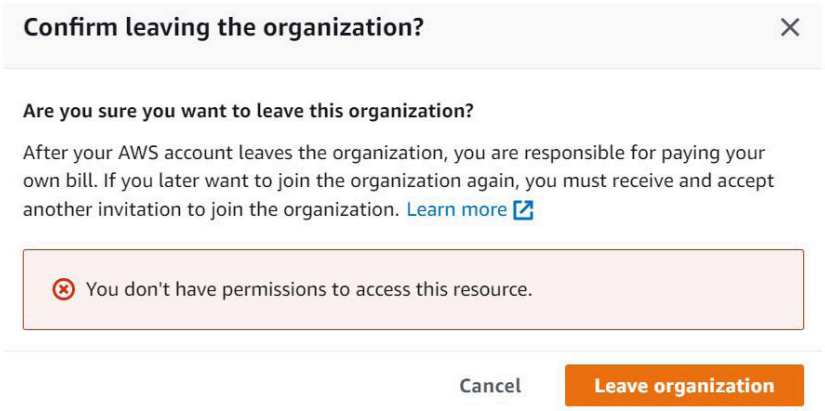


Figure 1.20 – SCPs prevent the root user of DevAccount from leaving the organization

As you can see, even the root user of the **Todo Plus Development** account can now not leave the organization. Ultimately, this policy will ensure that only the administrator of the management account can control the membership of member accounts.

Now that you have configured your AWS accounts and defined an AWS organization, you are ready to start working on the project tasks in the rest of this study guide. However, before proceeding with the project tasks, it is strongly recommended that you create a billing alarm to notify you if you cross the recommended budget of 25 USD in your exercises. It is advised that you complete the exercises in this book as quickly as possible so that you do not incur unnecessary charges. Once you have completed the project exercises and tested the solution, you must terminate the resources you deploy to avoid incurring further costs.

Project Task 1.8 – Creating a Billing Alarm

In this project task, you will create a billing alarm to notify you if cross the 25 USD spend budget:

1. Ensure that you are logged in to the management account.

2. In the top-right corner of the screen, click on **Todo Plus Management** and select **Billing and Cost Management**, as per the following screenshot:

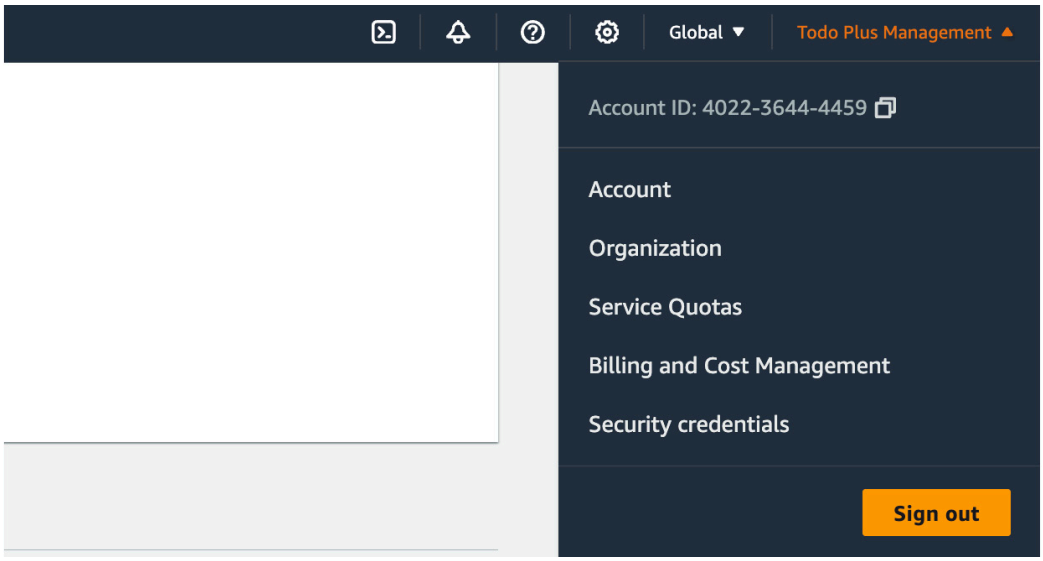


Figure 1.21 – Accessing the Billing and Cost Management console

3. In the **Billing and Cost Management** console, select **Billing Preferences** from the left-hand menu under **Preferences and Settings**.
4. Click the **Edit** button in the **Alert preferences** section, as per the following screenshot:

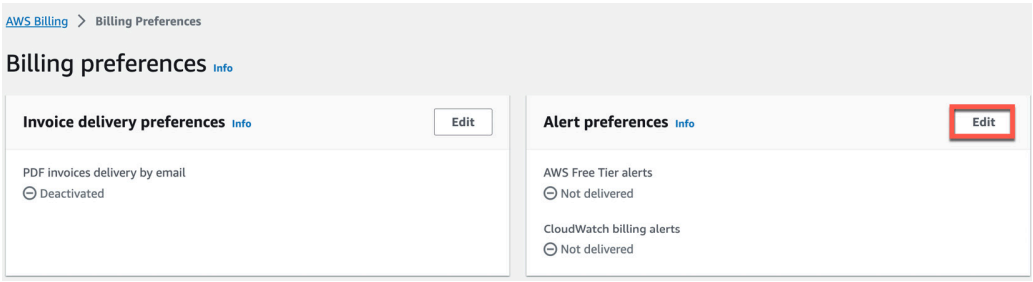


Figure 1.22 – Alert preferences setting

5. Ensure that you select the checkbox next to **Receive CloudWatch billing alerts** and click **Update**.
6. Using the top search bar, search for **`CloudWatch`** and navigate to the **Amazon CloudWatch** console.
7. In the CloudWatch console, from the left-hand menu, click **Billing** under **Alarms**.

8. Click the **Create alarm** button and you will be presented with a four-step wizard to create your billing alarm:
    1. Step 1: **Specify metric and conditions** – In this step, ensure that **Static** is selected under **Threshold type**. Next, select **Greater** as the alarm condition in the **Whenever EstimatedCharges is...** section, as per the following screenshot. Also, set the **USD** amount to **25**.

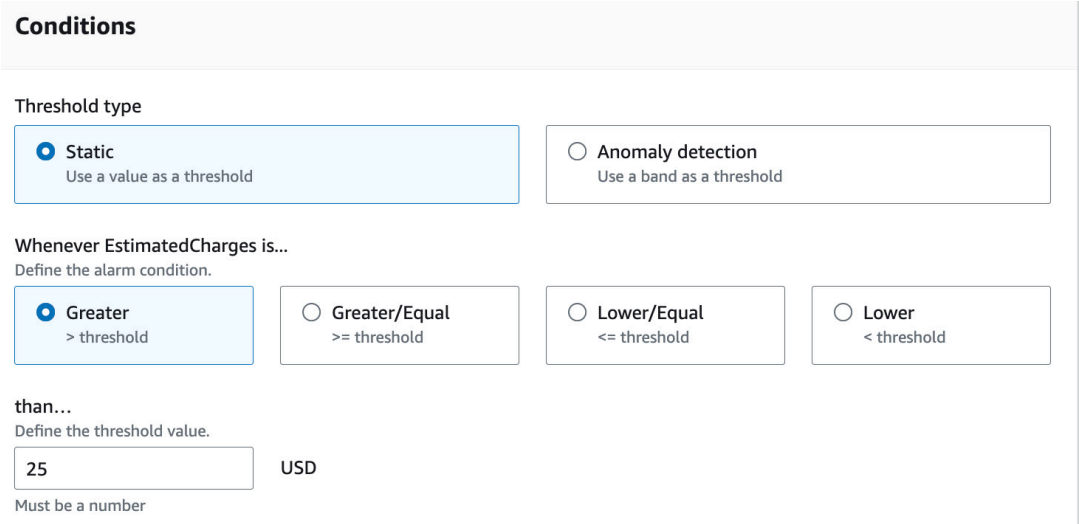


Figure 1.23 – Configuring billing alarm conditions

Click **Next**.

    2. Step 2: **Configure actions** – In this step, select **In alarm** under **Alarm state trigger**.

Click **Create a new topic**.

Provide a descriptive name under **Create a new topic...**. For example, you can provide a name such as **`TodoPlus-Billing-Alarm`**.

Provide an email address where you would like to receive the alert, and then click **Create topic**, as per the following screenshot:

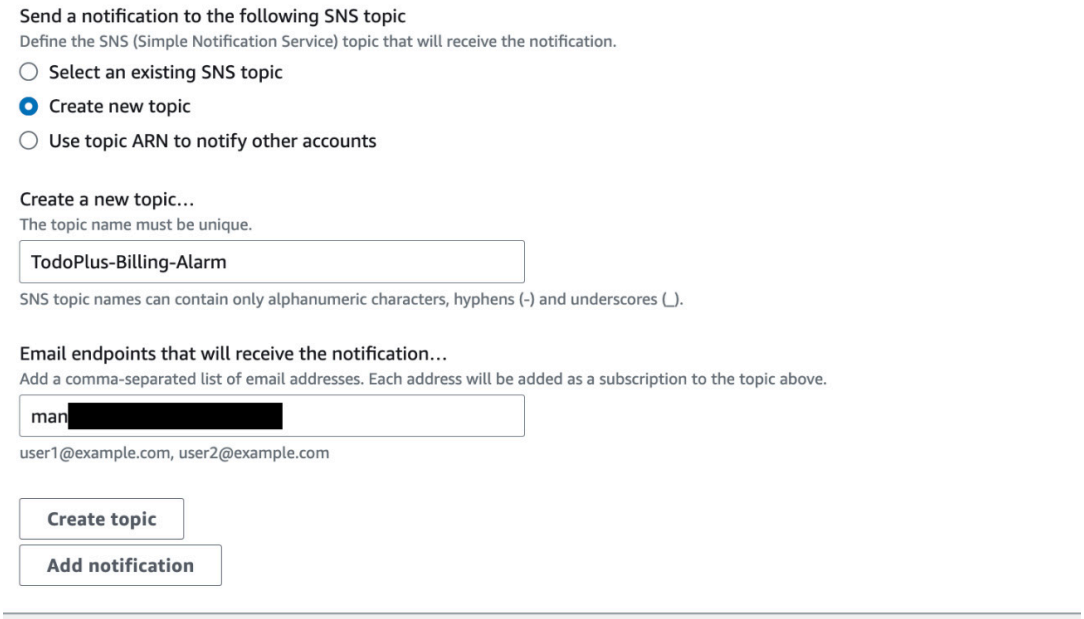


Figure 1.24 – Creating a notification topic and defining an email endpoint

Scroll to the bottom of the screen and click **Next**.

3. Step 3: **Add name and description** – In this step, you will define the alarm name and an optional description.

Click **Next**.

4. Step 4: **Preview and create** – In this step, you will review the settings and then click the **Create alarm** button at the bottom of the screen.

9. Although you have now created the alarm, you will not be able to receive the email alerts until you verify your subscription to the notification topic. To do this, you must log in to the email account that you specified in the previous steps and confirm your subscription.

In the next section, we will provide a summary of this chapter.

Summary

This chapter discussed the importance of a multi-account architecture, examining the vast use cases for setting up and managing multiple AWS accounts. It falls within *Domain 2: Security* in the *AWS Certified Developer Associate* exam with a particular focus on the task statement, *Implement authentication and/or authorization for applications and AWS services*. Specifically, you learned that hosting different workloads and even different environments in separate AWS accounts is essential for ensuring higher levels of security and reducing the impact of mishaps in one environment affecting workloads in another. With multiple AWS accounts, you can separate individual applications and even have a separate account for their development lifecycle.

With multiple AWS accounts, you can also separate internal applications and workloads from customer-facing ones, further enhancing security. In this chapter, you learned how to manage AWS accounts using AWS Organizations. You learned how to apply service control policies to prevent member accounts from specific services, resources, and regions. The management account in an AWS organization has the highest authority, and you can even prevent administrators of member accounts from leaving the organization.

Although AWS Organizations offers the ability to build guardrails for your member accounts, who or what can work with those permitted services and make API calls still needs to be defined using AWS IAM or by defining resource-based policies. We perform a deep-dive study of the IAM services in the next chapter.

Further Reading

The following are must-read resources to understand the approaches you can use in the real world when designing a multi-account architecture for your business and clients:

- *Organizing Your AWS Environment Using Multiple Accounts*: **https://docs.aws.amazon.com/whitepapers/latest/organizing-your-aws-environment/organizing-your-aws-environment.html**
- *Service Control Policy Examples*: **https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scps_examples.html**

Exam Readiness Drill – Chapter Review Questions

Apart from a solid understanding of key concepts, being able to think quickly under time pressure is a skill that will help you ace your certification exam. That is why working on these skills early on in your learning journey is key.

Chapter review questions are designed to improve your test-taking skills progressively with each chapter you learn and review your understanding of key concepts in the chapter at the same time. You'll find these at the end of each chapter.

HOW TO ACCESS THESE MATERIALS

To learn how to access these resources, head over to the chapter titled ***Chapter 17****, Accessing the Online Resources.*

To open the Chapter Review Questions for this chapter, perform the following steps:

1. Click the link – **https://packt.link/DVAC02_CH01**.
   Alternatively, you can scan the following **QR code** (*Figure 1.25*):

Figure 1.25 – QR code that opens Chapter Review Questions for logged-in users

2. Once you log in, you'll see a page similar to the one shown in *Figure 1.26*:

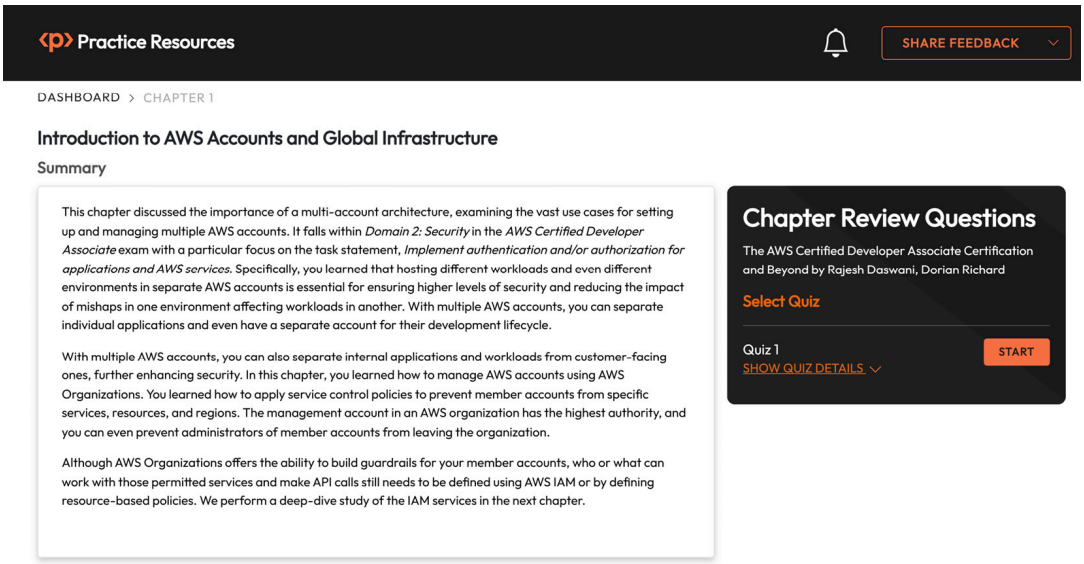


Figure 1.26 – Chapter Review Questions for Chapter 1

3. Once ready, start the following practice drills, re-attempting the quiz multiple times.

Exam Readiness Drill

For the first three attempts, don't worry about the time limit.

ATTEMPT 1

The first time, aim for at least **40%**. Look at the answers you got wrong and read the relevant sections in the chapter again to fix your learning

gaps.

ATTEMPT 2

The second time, aim for at least **60%.** Look at the answers you got wrong and read the relevant sections in the chapter again to fix any remaining learning gaps.

ATTEMPT 3

The third time, aim for at least **75%.** Once you score 75% or more, you start working on your timing.

TIP

You may take more than **three** *attempts to reach 75%. That's okay. Just review the relevant sections in the chapter till you get there.*

Working On Timing

Target: Your aim is to keep the score the same while trying to answer these questions as quickly as possible. Here's an example of how your next attempts should look like:

Attempt	Score	Time Taken
Attempt 5	77%	21 mins 30 seconds
Attempt 6	78%	18 mins 34 seconds
Attempt 7	76%	14 mins 44 seconds

Table 1.1 – Sample timing practice drills on the online platform

NOTE

The time limits shown in the above table are just examples. Set your own time limits with each attempt based on the time limit of the quiz on the website.

With each new attempt, your score should stay above **75%** while your "time taken" to complete should "decrease". Repeat as many attempts as you want till you feel confident dealing with the time pressure.

Previous chapter
< Preface

Next chapter
Chapter 2: Securing Access with AWS Identity and Access Management >