Part 1: Introduction to IaC

In this first part of the training walk through, you will have some common tasks to complete, then you will be able to choose if you want to complete the training with Bicep or Terraform (or both).

First, let's take a minute to get familiar with Azure and how to work with various scopes for management and deployment.

Basics of Azure deployment, RBAC, and Policy scopes

Before getting started, it's important to note that there are scopes for executing deployments at Azure. The following scopes are available:

- management group
- subscription
- resource group

Management Groups

Management groups can have multiple subscriptions, so they are the top-level scope. Anything done at the management group level can 'trickle-down' to the subscriptions and resource groups below it. For example, if you assign a policy at the management group level, it will apply to all subscriptions and resource groups below it. If you create the policy at the management group but don't apply it, it will not apply to the subscriptions and resource groups below it unless they directly implement the policy.

Only some resources make sense to deploy at this level. For example, you typically deploy policies at this level. You may also deploy things like Azure Monitor Workspaces, Azure Security Center, etc. You would not deploy things like storage accounts, virtual machines, etc. at this level, as those are typically scoped to a subscription in a resource group.

Management groups can be nested within other management groups as well. For example, you may have a management group for your entire organization, and then you may have management groups for each department, and then you may have management groups for each team within the department. This is a good way to organize your resources and deployments, but it is not required, and this is typically not done unless you have a large organization with many teams and/or departments.

Subscriptions

Subscriptions are the next level down from management groups. Subscriptions are an excellent barrier for RBAC, policy, and billing so at this level, resources are typically mapped to environments and/or clients. You can have multiple subscriptions under a single management group. You can also have multiple resource groups under a subscription. For example, common solutions for a single organization may be to have a subscription for each environment (dev, prod). If your company has multiple clients, you may have a subscription for each client, and you many even have multiple subscriptions for each client (for the client dev, prod, etc).

Resource Groups

Resource groups are a barrier for resources that are typically grouped together for lifecycle and/or security purposes. Resource groups are really for us humans that can't keep things straight in our heads over multiple workloads. Azure will allow you to group in resource groups but that does not limit you to what can be deployed or to which region the resources can be deployed. For example, you can deploy a storage account in a resource group in the East US region and a virtual machine in the same resource group in the West US region. This is not recommended, but it is possible. Typically, it would be recommended to separate workloads such as your web application with it's database, virtual machines, keyvaults, etc into groups that make sense both from a security and lifecycle perspective.

For example, RBAC controls can let a client view resources only in one single resource group where your team might need to see all of the groups (subscription-level permissions).

All resources live in resource groups, so it will be up to your team on how you want to deploy them.

Why is this important?

The reason this is important is because you can run deployments that only target a single resource group or you can run a deployment that targets the subscription and can therefore span multiple resource groups.

For example, suppose you just need a single storage account. You can do that deployment to a single resource group. Your IAC principal then only needs permission to that resource group (we'll do this in the first activity).

However, in real deployments for your company, you'll likely see a deployment that spans your entire subscription. For example, you may have a deployment that creates a resource group for your web application and its resources, a resource group for your KeyVault used to encrypt storage and database keys, and in more robust scenarios you may also need to deploy networks and resources related to the networking. In these cases, you'll need to have permissions to the entire subscription, and you will likely create an orchestrator file (just a bicep or terraform file that calls other bicep or terraform files) that will deploy all of the resources with one deployment operation calling to modules. We will do this in part two when we deploy the entire application in part 2.

Task 1: Complete the IaC activity with Bicep or Terraform

Now that you have a resource group to deploy resources to, you can complete the IaC activity. You can choose to complete the activity with Bicep or Terraform.

Choose your path and complete the work using the tool of your choice. You can do both if you want to, but you only need to do one to complete the training.

> **Important**: There will be requirements for a unique name. When this happens, use the YYYYMMDD of today along with your initials. For example, if today is 2025-08-15 and your initials are `acw`, your unique identfier would be `20250815acw`. Append your unique identifier to the end of variables like the storage account name. This will ensure that you don't have naming conflicts with other people in the workshop. For example, if the storage account is named `mystorage` then your actual storage account name should be `mystorage20250815acw`.

1. Complete Part 1 - Introduction to IaC - Bicep

- or -

1. Complete Part 1 - Introduction to IaC - Terraform