Overview
On August 3, 2022, the Microsoft Tech Community hosted an Azure Deployments AMA. The live hour of Q&A provided members the opportunity to ask questions and voice feedback with the product team. We hope you join us live next time!

Resources

Azure.com

Azure Blog

Azure Updates -
Get the latest updates on Azure products and features

Microsoft Customer Co-creation -
Share your thoughts and influence the outcome before a single line of code is written. Play an early role in product and service development by helping Microsoft build and test the features that you need in Cloud and AI services.

Azure Documentation

Azure on Microsoft Q&A -
Find answers to your technical questions about selected Azure products.

Azure on Microsoft Learn

Azure Code Samples

Microsoft Mechanics

Azure Partner Hub

Azure Active Directory Hub

Microsoft Security and Compliance Hub

Introduction

Welcome to the Azure Deployments AMA! View the list of introductions in this thread.

General Discussion

Q: *Is there a good reference or best practice recommendation on how to organize IaC code in repositories in regards to pure deployment code and configuration/parameter code (split?) (link)*

A: IDK know of a good reference or hard/fast rule. A few things to consider:

never put secrets in source control - so if you have params that are secret, you need another store for those (GitHub, azdo, keyvault)

do you want [all] param/config changes to go through the same "test" that your deployment code does (e.g., do you want validate parameters in a pipeline or test) - and in some cases you may specifically *not* want this

I see a mix of both in practice - some params are all on source control, others use pipeline/deploy time variables.

Q: *I recently earned AZ 204 certification for Azure Developer. Can you please introduce me to these rollouts and biceps please? Is there any certification available for them? (link)*

A: As of now there's no certificates available specifically for the ARM deployments products (Bicep, Terraform, ARM Templates). We have a great learning path on MS Learn to get started and deep dive into Bicep here: https://docs.microsoft.com/en-us/azure/azure-resource-manager/bicep/learn-bicep

Q: *How far are we from having parity across ARM Templates, Bicep and Terraform? Right now we struggle with some deployments of Bicep and Terraform because they lag behind ARM. (link)*

A: Parity is certainly a journey - and we're building more generic capabilities as a stop gap. TF has the AzAPI provider that will allow you to access anything in the platform. Bicep allows for nested deployments/modules that help. That said - we'd love specific feedback on where you're running into parity problems, it helps prioritize.

Can you please explain further the lag that you have been experiencing with Bicep behind ARM?

As far as Terraform is concerned, we have released the AzAPI provider which, like Bicep, supports all Azure services including preview services on day 0.

Today, there is parity between ARM Templates and Bicep. Any resource that can be deployed in ARM can be deployed via Bicep. While there can be some lag in availability in the AzureRM provider in Terraform, we recently introduced the AzAPI provider for terraform which also has the same parity between ARM Templates/Bicep/TF: https://registry.terraform.io/providers/azure/azapi/latest/docs

Q1: *Like Synapse deployment - as of a few months back this was not possible from our efforts. Workspaces, VNETs, Linked Services, etc.*

A1: We would need more details about what is not working with these resource types in Bicep. There should not be any gap between these two tools. It might be easier to file an issue in the bicep GitHub repo so you can provide specific code samples, error details, etc. https://github.com/Azure/bicep/issues

Q: *Is there a document which can help to understand the core functions & configuration part? (link)*

A: Can you expand a bit more on what you mean by core/config?

Q1: I meant to in brief document on functions of bicep.

A1: We've made a lot of content for Bicep available in MS Learn. Here is the intro content, but there is a lot more available: https://docs.microsoft.com/en-us/learn/paths/fundamentals-bicep/

This may be the simplest, single source of truth: https://docs.microsoft.com/en-us/azure/azure-resource-manager/bicep/bicep-functions The template reference docs have both JSON and bicep syntax: https://docs.microsoft.com/en-us/azure/templates/

Q: *I learned Azure Basics and want to start scripting AzDeployment. What is an easy way for new rollouts, biceps? And by what technique/tool it is easy to manage or alter an environment. Should it be Infrastructure by code completely? (link)*

A: For getting started, some people find it easier to use Az CLI or Azure PowerShell. They tend to be more familiar for those with IT backgrounds and have done PowerShell or bash scripting. The issue with those tools is they tend not to scale well as your deployments get more complex -- they deploy resources serially instead of in parallel for example.

If you want to use one of the declarative tools, then both Terraform and Bicep are great options. We offer some introductory guides for both tools, which I've linked to below:

* Bicep: https://docs.microsoft.com/en-us/learn/paths/fundamentals-bicep/

* Terraform: https://learn.hashicorp.com/collections/terraform/azure-get-started

Q1: *I did not have a look at the tools yet. I think of deploying and maintaining environments as an MSP. Or maybe rollout a PoC. We want to enhance our PowerShell-rollouts by Bicep or Terraform. The structure of an AVD environment is quite similar. Which one to concentrate on as a newbie?*

A1: You can't go wrong with either Bicep or Terraform. There are both pros and cons to either choice. You can read more about the comparison here: https://docs.microsoft.com/azure/developer/terraform/comparing-terraform-and-bicep?tabs=comparing-bicep-terraform-integration-features

Q: *Is it still useful to add biceps warnings due to broken API schemas to https://github.com/Azure/bicep/issues/784? I really would like to get to a stage where bicep can have a "Treat Warnings As Errors" setting. Right now, that doesn't work for obvious reasons. (link)*

A: Have you seen the "#disable-next-line" directive? You can use that to tell bicep to turn off specific warnings or errors and that may help you rely more on specific warnings and treat them as errors. It is still useful to add type issues to that thread. We are planning to revamp how we take in type issues, but in the meantime, we would like issues filed there.

Q: *I am looking for a good solution to multiple (1-200) Single Tenant Deployments instead of just using Environments with GitHub. Although that is the current solution.*

This page has been abandoned:

https://github.com/Azure/Multi-tenant-Landing-Zones

And the "Deploy All Customers" is not robust enough:

https://github.com/Azure/Multi-tenant-Landing-Zones/blob/main/.github/workflows/Deploy-All-Customer.yml

I'd love to hear some good examples of anyone else doing this.

(link)

A: Are you working for a managed service provider (MSP)? Can you give an example of what you would like to achieve?

Q1: *We've dealt with them in the past and they don't offer any good suggestions. What would be the best way to keep track of deploying multiple tenants? Or buzzwords to include in our monthly conversation with our Microsoft rep?*

A1: Are you looking to run a small PowerShell / Azure CLI script in 200 subscriptions? Or deploy a Bicep / Terraform template to each of them? As an ad hoc thing, or within a pipeline? Do you need to check that it has run well once? Or show that each of the 200 subscriptions is now compliant?

Q2: *Using Bicep to Onboard Subscriptions as part of a Pipeline. Using Bicep to Maintain existing services in those subscriptions as part of a Pipeline.*

A2: And are they projected into your tenant using Azure Lighthouse (so you have a pan multi-tenant view)? Or do you have a service principal with Contributor role in each of those subs?

Q3: *Service Principal with limited scope for each service in the subscription. Are you seeing customers switch over to Azure Lighthouse for SaaS deployments?*

A3: The good news is that you can absolutely use the same service principle in the MSP tenant and stamp out a cookie cutter environment in multiple customer subscriptions projected via Azure Lighthouse. I have a reference cookie-cutter repo for you.

The downside is that you cannot currently use for_each to elegantly loop through the environments. The open issue includes a nice terragrunt example to auto-generate the .tf files.

You could also do much the same with a service principle that is created in each customer tenant. Store those details in one or more key vaults and that is a workable variant of this. (Set the correct tenant_id. client_id and client_secret per customer provider alias block.)

Q: *Anyone have a list of great opensource tools they are using for their pipelines?*

static code analysis for IaC

https://github.com/bridgecrewio/checkov

static code analysis for GitHub Actions

https://github.com/rhysd/actionlint

kitchen sink

https://github.com/oxsecurity/megalinter

(link)

A: ARM Template Toolkit:

https://aka.ms/arm-ttk

ARM Template Analyzer
https://github.com/Azure/template-analyzer - more security focused - this is still in a "preview" state.

Q: *Is Microsoft going to broaden their Azure Quickstarts to include both Bicep and Terraform configurations in their gallery and GitHub repository? Is there a chance that Marketplace deployments might accept Terraform instead of solely ARM? (link)*

A: First, thanks for your interest in Terraform on Azure. We do have a set of Quickstarts for Terraform that are hosted on our Terraform GitHub repo: https://github.com/azure/terraform. This repo is nowhere near as extensive as the ARM repo but it is something that we are investing in and expect to grow over the short term. As you can see, this. repo is public so if you have any good examples that you would like to share, please feel free to submit a pull request. We will also look into whether there is a way to have a common set and list/index of Quickstarts across

ARM, Bicep and Terraform. As far as the market place goes, we do not currently have plans to add Terraform support here but it is something that we will investigate for the future.

In case you're referring to the Quickstart here: https://github.com/Azure/azure-quickstart-templates

Which are also here: https://azure.microsoft.com/en-us/resources/templates/ - but that UI is being migrated to here: https://aka.ms/azqst

We have bicep code for about 300 of the 1000 samples... we're going through the process of adding more. To see the bicep ones specifically click the aka link above and then check the bicep filter to see them.

That’s a wrap!

Thank you for joining for this fun hour! We hope you'll continue to ask questions and share your feedback.

See you next time!