# Adding a New Provider to Your Configuration

**Ned Bellavance**

MICROSOFT MVP, CLOUD AND DATACENTER MANAGEMENT

@ned1313 www.nedinthecloud.com

# Overview

New scenario requirements

Examine available providers

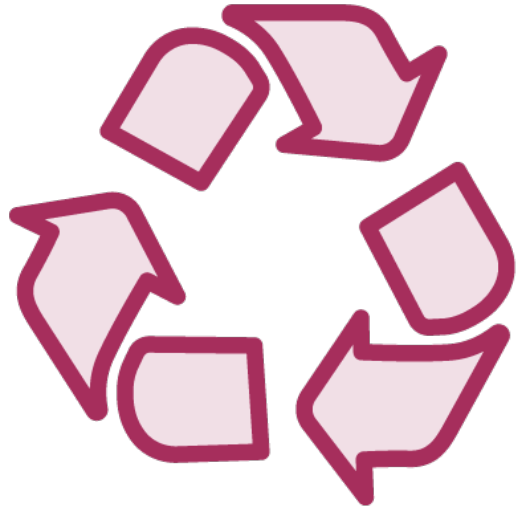
Explore functions and usage

# Automating Infrastructure Deployment

Provisioning resources

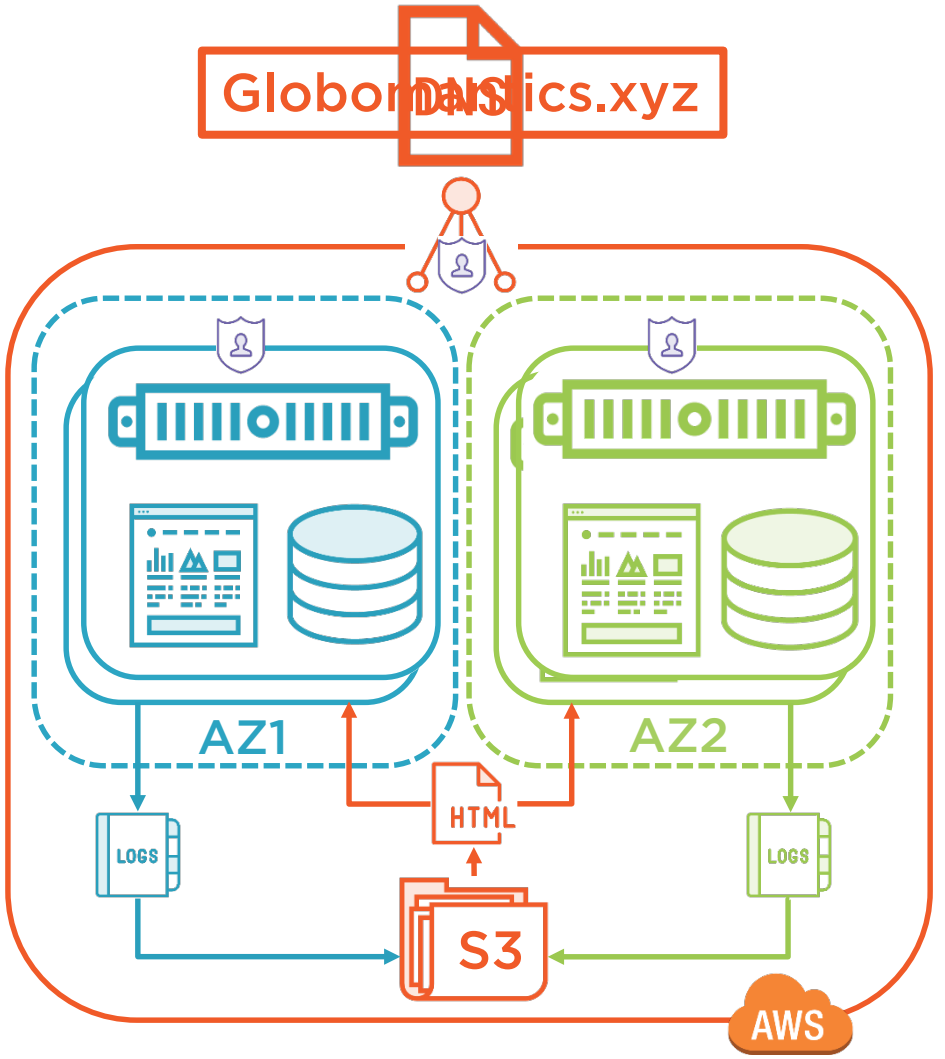
Planning updates

Using source control



Reusing templates

# The Scenario



# Terraform Providers

**IaaS, PaaS, and SaaS**

**Community and HashiCorp**

- AWS, Azure, GCP, and Oracle

**Open source using APIs**

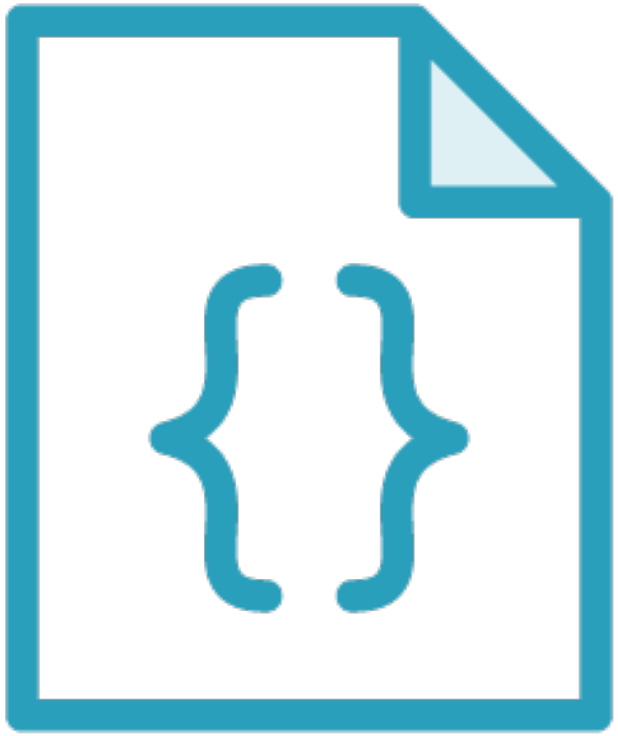
**Resources and data sources**

**Multiple instances**

# Provider Example

```
provider “azurerm” {
  subscription_id = “subscription-id”
  client_id = “principal-used-for-access”
  client_secret = “password-of-principal”
  tenant_id = “tenant-id”
  alias = “arm-1”
}
resource “azurerm_resource_group”{
  name = “resource-group-name”
  location = “East US”
  provider = “azurerm.arm-1”
}
```

# Terraform Functions



HashiCorp configuration language

Conditionals and functions

Types and values

Common uses

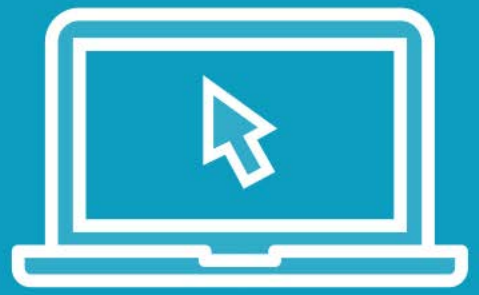
# Terraform Functions

```
#Configure networking
variable network_info {
 default = “10.0.0.0/8” #type, default, description
}
#Returns 10.1.0.0/16
cidr_block = ${cidrsubnet(var.network_info, 8, 1)}
#Returns 10.2.0.0/16
cidr_block = ${cidrsubnet(var.network_info, 8, 2)}
```

# Terraform Functions

```
#Create ami map
variable “amis” {
  type = “map”
  default = {
    us-east-1 = “ami-1234”
    us-west-1 = “ami-5678”
  }
}
ami = ${lookup(var.amis, “us-east-1”)}
```

# Demo



**Examine the Terraform file**

**Deploy the configuration**

**Review the results**

**Play along!**

- AWS account
- Azure subscription
- DNS domain
- Terraform software (terraform.io)
- Demo files

Some of the resources deployed in AWS and Azure may cost **money**. You've been warned.

# Summary

**Terraform providers**

**Functions**

**Terraform.exe commands**

**Coming up**

- Managing variables
- Multiple environments
- Securing credentials