Bee-a-Learner Follow
Oct 23, 2021 · 3 min read · · Listen

Deploying API management with terraform

Today in this blog I will share how to configure Azure API management with terraform.

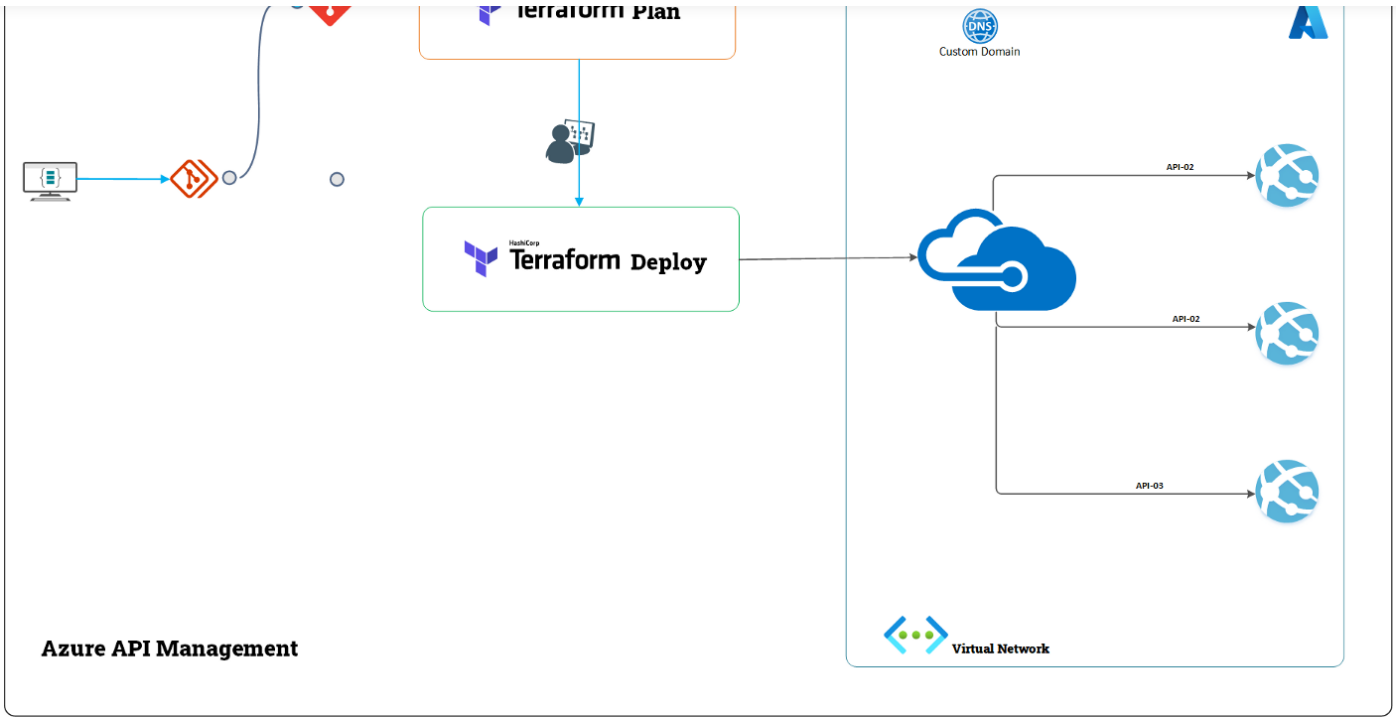
Before this I would like to give you some background about the API Management service.

What is API Management ?

According to Azure documents "*API Management (APIM) is a way to create consistent and modern API gateways for existing back-end services*".

This service offers so many other features to host your API using API management.

In this blog I will show you how to automate azure API management with terraform, following will be the terraform deployment should looks like



API Management deployment flow

first of all we are going use terraform azure provider and azurerm_api_management, here in this resource block we are going to define API management

```
1
2   resource "azurerm_api_management" "apim" {
3       location                  = var.location
4       name                      = var.apim_name
5       publisher_email           = var.publisher_email
6       publisher_name            = var.publisher_name
7       resource_group_name       = var.resource_group_name
8       sku_name                  = var.sku_name
9       tags                      = var.tags
10      virtual_network_type      = var.virtual_network_type
11
12
13     identity {
14           type         =   "SystemAssigned"
15      }
16
17      protocols {
```

```
23
24          terms_of_service {
25              consent_required = false
26              enabled          = false
27          }
28      }
29
30      dynamic "virtual_network_configuration" {
31          for_each = var.virtual_network_type=="Internal"?[1]:[]
32          content {
33              subnet_id = var.apim_subnet_id
34          }
35      }
36
37      lifecycle {
38        ignore_changes = [hostname_configuration]
39      }
40  }
41
42
```

apim.tf hosted with ❤ by GitHub view raw

Here in the api management resource block we are configuring API management as internal or external and that's the reason we have virtual network configuration block

- If We would like our API management to configure custom domain then you need to pass the parameter '*requires_custom_host_name_configuration*' value as '***true***' following is the code to configure custom domain:

```
1  data "azurerm_key_vault" "keyvault" {
2    count               = var.requires_custom_host_name_configuration ? 1 : 0
3    name                = var.wildcard_certificate_key_vault_name
4    resource_group_name = var.wildcard_certificate_key_vault_resource_group_name
5  }
6
7  data "azurerm_key_vault_certificate" "cert" {
```

```
resource "azurerm_key_vault_access_policy" "keyvault" {

  count                  = var.requires_custom_host_name_configuration ? 1 : 0
  key_vault_id           = data.azurerm_key_vault.keyvault[0].id

  tenant_id              = azurerm_api_management.apim.identity[0].tenant_id
  object_id              = azurerm_api_management.apim.identity[0].principal_id

  secret_permissions = [
    "Get",
    "List"
  ]
}

resource "azurerm_api_management_custom_domain" "apim_domain" {

    count                = var.requires_custom_host_name_configuration ? 1 : 0
    api_management_id    = azurerm_api_management.apim.id

        dynamic "developer_portal" {
            for_each = var.developer_portal_host_name != "" ?[1]:[0]
                content {
                    host_name                    = var.developer_portal_host_name
                    key_vault_id                 = data.azurerm_key_vault_certificate.cert
                    negotiate_client_certificate = false
                }
        }

        dynamic "management" {
            for_each = var.management_host_name != "" ?[1]:[0]
                content {
                    host_name                    = var.management_host_name
                    key_vault_id                 = data.azurerm_key_vault_certificate.cert
                    negotiate_client_certificate = false
                }
        }

        dynamic "proxy" {
            for_each = var.proxy_host_name != "" ?[1]:[0]
```

```
        }

    lifecycle {
        ignore_changes = [
            proxy
        ]
    }

    depends_on = [ azurerm_api_management.apim, azurerm_key_vault_access_policy.keyvault ]
}
```

apim-custom-domain.tf hosted with ❤ by GitHub view raw

Here we are using keyvault block to read the certificate so that the certificate can be used to setup custom domain.

I am also using API management policy resource block in case if you would like to configure API management policies at API level globally, product or API level depends on the xml configuration which you will provide

```
resource "azurerm_api_management_policy" "policy" {
  count = var.apim_default_policy_path==""?0:1
  api_management_id = azurerm_api_management.apim.id
  xml_content       = file(var.apim_default_policy_path)
}
```

apim-policy.tf hosted with ❤ by GitHub view raw

That's all you need to setup Private or external type of API management with optional custom domain and policy setup, here is all the variable you might requires to pass the values to run the code

```
variable "apim_name" {
  type          = string
```

```
  type          = string
  description   = "api management resource_group_name"
}

variable "location" {
  type          = string
  description   = "api management location"
  #default       = "westeurope"
}


variable "sku_name" {
  type          = string
  description   = "api management sku"
  default       = "Developer_1"
}

variable "publisher_name" {
  type          = string
  description   = "api management publisher neam"
}

variable "publisher_email" {
  type          = string
  description   = "api management publisher email"
}



variable "apim_user_assigned_identity" {
  type          = string
  description   = "api management apim_user_assigned_identity"
  #default       = "dev-portal.nonprod.contoso.com"
}

variable "developer_portal_host_name" {
  type          = string
  description   = "api management developer portal host name"
  #default       = "dev-portal.nonprod.contoso.com"
}
```

```
54    #default       = "management-dev.nonprod.contoso.com"
55  }
56
57
58  variable "proxy_host_name" {
59    type          = string
60    description   = "api management, proxy host name"
61    #default       = "dev-api.nonprod.contoso.com"
62  }
63
64  ################# Custom domain name configuration ############################
65  variable "requires_custom_host_name_configuration" {
66    type          = bool
67    description   = "true if requires custom host name configuration, otherwise false (defa
68    #default       =  true
69  }
70
71  variable "wildcard_certificate_key_vault_name" {
72    type          = string
73    description   = "keyvault name which holds a certificate to configure apim custom domai
74  }
75
76
77  variable "wildcard_certificate_key_vault_resource_group_name" {
78    type          = string
79    description   = "resource_group name of keyvault which holds a certificate to configure
80  }
81
82  variable "wildcard_certificate_name" {
83    type          = string
84    description   = "keyvault certificate name which will be used to configure apim custom
85  }
86
87  ##############*****************************************************************************
88
89  variable "virtual_network_type" {
90    type          = string
91    description   = "api management virtual network type"
92    default       = "Internal"
```

```
 99    type          = string
100    description   = "api management virtual network subnet id, (requires if APIM network ty
101  }
102
103
104  variable "apim_default_policy_path" {
105    type = string
106    description = "(optional) api management default policy path, if any policy needed to b
107    default = ""
108  }
109
110  variable "tags" {
111    description   = "api management resource tags"
112
113    default       = {
114          "Data_Classification" = "Standard"
115      }
116  }
```

apim-variables.tf hosted with ❤️ by GitHub view raw

terraform also has some additional resource blocks related to API management such as configuration of product, policies, users, api import which you can configure with in you API management resource block, but this is something in most of the project are dependent of application source deployment and I wont recommend to setup those as part of Infrastructure deployment unless you have some static setup of API configuration which you would like to deploy.

following is the sample API management terraform variables which you can use to configure the resource in your azure subscription

Additionally if you want to learn more about API management, I have a lots of videos on my YouTube channel which you watch.

References:

GitHub repository like to find all the code:

GitHub - bee-a-learner/terraform: code block of terraform

code block of terraform. Contribute to bee-a-learner/terraform development by creating an account on GitHub.

github.com

terraform document:

https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/api_management

MS Docs:

Azure API Management overview and key concepts

API Management (APIM) is a way to create consistent and modern API gateways for existing back-end services. API...

docs.microsoft.com