**Terraform import**

import 是在 terraform root command，用來 import 已經存在的 remote resource 到 state。可以看 [官方文件 import](https://www.terraform.io/docs/cli/import/index.html) 中描述

* import 已經存在的 infrastructure (remote resource) 將他納入 terraform 的管理中
* 最常見的用例是，在一個還沒導入 terraform 的專案中，已經有 infra 存在時，逐漸導入 terraform 時，陸續 import 到 terraform

要 import 之前，要先手寫產生 .tf resource

* ex. 如果要 import subnet[0]，要先寫 subnet[0] 的 resource 在 .tf 中
* 然後使用新的 subnet[0] 的 resource address，將 remote resource import 近來

文件提到 Terraform 目前的 import 實作，只影響 state，不會產生 .tf resource。這是什麼意思？

* 要 import 已經存在的 infra，卻還要先寫 .tf resource 再 import，這樣不是很麻煩

另一個專案 [Google Cloud Platform 推出的 terraformer](https://github.com/GoogleCloudPlatform/terraformer) 有提供直接 import 並產生 .tf 檔案的方法

* 留在 state 講解完後再跟大家分享，這邊先不提

**import example**

這邊接續上堂課 state rm 的範例，直接使用 import

* 注意這不是實務上常出現的例子，理想中你不會有一個隊友，沒事 state rm 東西

上堂課我們卡在這邊

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.

Only 'yes' will be accepted to approve.

Enter a value: yes

module.network.azurerm\_subnet.subnet[0]: Creating...

╷

│ Error: A resource with the ID "/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-1" already exists - to be managed via Terraform this resource needs to be imported into the State. Please see the resource documentation for "azurerm\_subnet" for more information.

│

│ with module.network.azurerm\_subnet.subnet[0],

│ on .terraform/modules/network/main.tf line 15, in resource "azurerm\_subnet" "subnet":

│ 15: resource "azurerm\_subnet" "subnet" {

│

╵

ERRO[0129] 1 error occurred:

\* exit status 1

剛好這例子中

* .tf resource / address 已經有了，就是 module.network.azurerm\_subnet.subnet[0]
* remote infra 也已經存在， ID "/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-1"

[到 terraform registry 中的各個 resource 文件下查詢 import 語法，例如 azurerm\_subnet](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/subnet#import) 底下有 import 語法

* 由於各家公有雲的 API 設計本質上就不同，因此不同公有雲的 import syntax 也不同，每次記得來 registry 文件查詢
* 根據文件的 import 語法，換成上面 error message 提供的 id
* error message 這麼剛好嗎？這是有先配好的，也就是預期 id / name 重複的 resource create error 時，要提供可以 import 的 id 在 error message 中
* 一樣記得 bash 中，該 double quote 記得要 quote
* 一樣記得我們是用 terragrunt shim layer 呼叫 terraform 的指令

terraform import azurerm\_subnet.exampleSubnet /subscriptions/00000000-0000-0000-0000-000000000000/resourceGroups/mygroup1/providers/Microsoft.Network/virtualNetworks/myvnet1/subnets/mysubnet1

cd azure/foundation/compute\_network

terragrunt import "module.network.azurerm\_subnet.subnet[0]" "/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-1"

module.network.azurerm\_subnet.subnet[0]: Importing from ID "/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-1"...

module.network.azurerm\_subnet.subnet[0]: Import prepared!

Prepared azurerm\_subnet for import

module.network.azurerm\_subnet.subnet[0]: Refreshing state... [id=/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-1]

Import successful!

The resources that were imported are shown above. These resources are now in

your Terraform state and will henceforth be managed by Terraform.

顯示 import 成功

* 一樣養成習慣，每次執行 state 變更後，務必檢查 state
* state list 檢查 address 狀態
* plan 檢查 .tf 與 state 是否符合

terragrunt state list

module.network.data.azurerm\_resource\_group.network

module.network.azurerm\_subnet.subnet[0]

module.network.azurerm\_subnet.subnet[1]

module.network.azurerm\_subnet.subnet[2]

module.network.azurerm\_virtual\_network.vnet

terragrunt plan

Changes to Outputs:

~ vnet\_subnets = [

- null,

+ "/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-1",

"/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-2",

*# (1 unchanged element hidden)*

]

You can apply this plan to save these new output values to the Terraform

state, without changing any real infrastructure.

plan 顯示剩下唯一有差異的地方是 output 的值，我們進行 apply 來更新這個 output

* state 變更後，有可能會影響到其他的 state，也就是會有連鎖反應
* 這邊只是影響一個 output，沒什麼差別，但在複雜的專案中，要特別小心

terragrunt apply

**Practical terraform import**

上面我們直接使用 state rm 後的例子來做 import 示範，但實務上會更接近

* 去 terraform registry document 找 import syntax，包含 id 或 path 的定位路徑
* 由於不一定 import state 會與 .tf resource 的設定完全符合，一般來說我們會在 import 後 plan 一次
* 根據 plan 有所出入，調整 .tf resource 直到符合 state (也就是 plan 時計算結果是 no changes)
* 手動完成 .tf resource 與 remote resource 的對照

**multiple collaborater workflow**

與前面一樣，更改 state 會有多人協作的問題，這邊操作流程類似。如果團隊在專案途中開始導入 terraform 的話，不妨參考一下這個流程

* master 是 auto plan + auto apply
* 開新的 branch，為 remote resource 增加 .tf resource，對應 remote resource
* 開 PR，進行 code review
* review 完成後，不要 merge 進 master 造成 auto apply failure，而是標上 do-not-merge 與 manual-operation-required 等 tag
* notify team 即將要進行手動的 terraform import，告知影響的 module，請團隊成員不要進來 plan / apply
* 開始手動操作 terraform import
* 完成 import 後，手動檢查 state list 與 terraform plan result
* 確定 .tf 與 state 與 remote resource 三方達成一致
* merge PR 到 master
* master auto plan 會顯示 no changes

**When you need import**

所以，何時需要 import

一個是團隊成員中有 state rm 狂人，或是有人搞砸弄壞 state 被迫上來修

* 實際操作流程如 state rm 再 import 例子

一個是上面第二個例子，就是逐漸導入 terraform 的過程，如果有既有的 resource，這樣會常常用到 import，請小心操作

另外一個，也算是常見的例子，就是被中斷的 terraform workflow

* 可能是 terraform apply 途中網路斷線，被迫停止
* 此時 public cloud API 已經打出去了，但在等待 resource 創建完成需要時間，卡在中途
  + 有 .tf resource，apply 到一半
  + 有 remote resource
  + state 尚未更新，也就是缺 state
* 這時的修復流程可能會用到 import
  + 首先透過 az-cli 或是 web console，先檢查 remote resource 實際狀況到底跟 .tf 差多少
  + force-unlock state lock，因為 apply 到一半沒有 state 收尾，仍然是 lock 狀態（unlock 之前最好跟 team member 確認一下）
  + 取得 id，進行 terraform import 修復
* 接續 apply 的工作