[Terraform 官方文件](https://www.terraform.io/docs/cli/state/taint.html) 已經依照 state manipulation 的使用情境做了分類，有以下幾個類別

* Inspecting State
  + state list
  + state show
  + refresh
* Forcing Re-creation (Tainting)
  + taint
  + untaint
* Moving Resources
  + state mv
  + state rm
  + state replace-provider
* Disaster Recovery
  + state pull
  + state push
  + force-unlock

**Terraform taint**

[Terraform 官方文件 taint](https://www.terraform.io/docs/cli/state/taint.html)

在 terraform 正常工作流程中，在不需要 destroy + create 的情形下，terraform 會盡量 update 計有的 resource。

然而，有時我們管理 infrastructrue 的時候，就是會需要強制 recreate 的動作，terraform 提供了 state taint / state untaint 來滿足需求

使用 taint 可以標記一個已經存在的 state address，在下次 plan / apply 時，terraform 會強制 replace 這個 state 相關的 resource

* .tf resource 不變
* 強制 replace remote resource
* 也同時 replace state

相同的 resource address 下，會刪除原先的 remote resource，並 create 一個全新的 remote resource，並把 state 放在原本的 resource address 下

**example**

一樣使用用到爛的 azure/foundation/compute\_network 示範

cd azure/foundation/compute\_network

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

Resource instance module.network.azurerm\_subnet.subnet[0] has been marked as tainted.

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

-/+ destroy and then create replacement

Terraform will perform the following actions:

*# module.network.azurerm\_subnet.subnet[0] is tainted, so must be replaced*

-/+ resource "azurerm\_subnet" "subnet" {

~ address\_prefix = "10.2.1.0/24" -> (known after apply)

~ id = "/subscriptions/6fce7237-7e8e-4053-8e7d-ecf8a7c392ce/resourceGroups/terraform-30-days/providers/Microsoft.Network/virtualNetworks/acctvnet/subnets/dev-1" -> (known after apply)

name = "dev-1"

- service\_endpoint\_policy\_ids = [] -> null

*# (6 unchanged attributes hidden)*

- timeouts {}

}

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

Changes to Outputs:

~ vnet\_subnets = [

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

+ (known after apply),

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

*# (1 unchanged element hidden)*

]

─────────────────────────────────────────────────────────────────────────────

* taint 掉 module.network.azurerm\_subnet.subnet[0]
* state list 仍然存在
* plan 的時候顯示 destroy and then create replacement

如果 apply，terraform 就會執行 destroy + create

* 過程中會影響依賴 subnet 的公有雲 resource
* 如果 subnet 上面沒有什麼東西，可以嘗試 apply 下去

**untaint**

[官方文件 untaint](https://www.terraform.io/docs/cli/commands/untaint.html)

* terraform 內部有 mark tained 的機制
* 如果一個 apply 途中，有多步驟的 create 中途出錯，terraform 會自動把產生到一半的 resource 加上 tainted 標記
* 因為 terraform 不能保證 create 到一半的 resource 功能是正常的
* 下次 apply 就會 destroy + create 全新的 resource，確保 resource 完整

如果此時想要人為介入，便可以使用 untaint 來移除 tainted 標記

* 移除後，下次 plan / apply 就不會 destroy + create resource

**scenario of taint**

* 上面提到的，apply 途中出錯，卡在中間，可以用 taint / untaint 控制下次 plan / apply
* provider 覺得不用 recreate，想要人為強制 recreate 的時候
* 如果 resource / module 中有使用 lifecycle { ignore\_change }，使用 taint 可以強制 resource update

**taint deprecation**

在 Terraform v0.15.2 之後，terraform 提供了新的方法來操作 taint

* 何必 taint 之後再 apply
* 直接 apply 時，指定 -replace="resource address" 就行了

terraform apply -replace="aws\_instance.example[0]"