Terraform Course

① Can save tf plan to a file

terraform plan -out <filename>
↓
infra.plan

② apply from planned file

- terraform apply infra.plan
- even if you changed infra but you want to run infra.plan ka infra → you can

③ terraform show to read contents of plan file since it's a binary file

---

terraform output <variable name>

→ command used to extract value of an

output variable from state file

terraform settings are used to configure project specific terraform behaviours, such as requiring a min of version to apply your config

```
terraform {
    required version =
    required providers = {

    }
}
```

Zipmap function constructs a map from a list of keys & corresponding list of values

```
zipmap ( keylist, valuelist)
```

Comments in terraform code

or // or /* */

Meta arguments → we use these args when we want to ignore any manual changes done through Aws console to a resource

basically we do not want to overwrite the manual changes done through Aws console to any resource

```
resource {
  lifecycle {
    ignore_changes = [ ]
  }
}
```

↑ list of prop names like tags

---

dif meta args

depends_on, count, for_each, lifecycle, provider

meta args inside lifecycle meta args

- create_before_destroy
- prevent_destroy
- ignore_changes
- replace_triggered_by

---

terraform graph command produces descriptions of relationships b/w objects in tf config using DOT language

---

Terraform Provisioners

→ not a part of newer version of exams (confirm with Rohan/ Jayati / Ritik)

Terraform Modules

(1) DRY - do not repeat yourself

(2) Tf modules allow us to centralize the resources config of it makes it easier for multiple projects to re-use tf code

(3) terraform init downloads the module code in local workspace

(4) tf registry can contain multiple modules for specific infra resource maintained by dff users

don't use any random modules since some hacker might have planted malicious code in a module

---

To make a module of say EC2, IAM we use resource block to refer a module we use module block which must contain a source argument that contains location to

referenced module

specify the version of module as well

git repo or local path or bitbucket

→ avoid hard coding of values as a part of modules as then later it will be difficult to modify in resource

↓

use variables

Module outputs

ensure to include o/p values in the module code for better flexibility & integration with other resources & projects

Root & child modules

Root module resides in main directory & is entry point of infra

child module is one that is called by another module

Requirements for publishing module

1. module must be on github & must be a public repo
2. must use 3 part name format

terraform-<PROVIDER>-<NAME>

3. repo desc = short desc of module
4. have standard module struct
5. uses tags to identify module versions

---

terraform workspace show

↓

to know WS you are in

terraform workspace { "new <name>"
delete
list
select }

Remote State Management

Alias meta argument

each provider can have 1 default config,
to include any no. of alt config
include extra name → alias

```
provider "aws" {
    alias = mumbai
    region = "ap-west-1"
}
```

Hashicorp Vault allows org to store secrets securely along with access

management for protecting secrets

Terraform Challenges

~~Q#1~~