

 Custom View Settings

Topic 1 - Single Topic

Question #1

Topic 1

The terraform.tfstate file always matches your currently built infrastructure.

- A. True
- B. False

Correct Answer: B

Reference:

<https://www.terraform.io/docs/language/state/index.html>

This state is stored by default in a local file named "terraform.tfstate", but it can also be stored remotely, which works better in a team environment.

Terraform uses this local state to create plans and make changes to your infrastructure. Prior to any operation, Terraform does a [refresh](#) to update the state with the real infrastructure.

The primary purpose of Terraform state is to store bindings between objects in a remote system and resource instances declared in your configuration. When Terraform creates a remote object in response to a change of configuration, it will record the identity of that remote object against a particular resource instance, and then potentially update or delete that object in response to future configuration changes.

Community vote distribution

B (96%)

4%

  **fabiomlop** Highly Voted 2 years ago

It doesn't necessarily always matches the current infrastructure, since one can manually change resources and therefore drift from the state configuration. Terraform has no way to know or track changes made outside of it.

upvoted 31 times

  **gargaditya** 1 year, 4 months ago

Incorrect, your next terraform plan or terraform refresh will inform you that there is change(provided the deployment was initially done via terraform).

tfstate reflects values from actual infrastructure deployed, and it picks drift from actual tf files.

upvoted 1 times

  **Nunyabiznes** 1 year, 3 months ago

Incorrect, question does not mention about "your so called " terraform plan or refresh commands

upvoted 3 times

 **hrajkuma** (Most Recent) 3 days, 15 hours ago

Selected Answer: B

upvoted 1 times

 **lmpz** 1 month, 3 weeks ago

is this exam valid on 21 May 2024?

upvoted 1 times

 **Hanu1** 5 months ago

Selected Answer: A

The terraform.tfstate file maintains the state of your infrastructure as managed by Terraform. It keeps track of the resources that Terraform has provisioned and their current state.

upvoted 1 times

 **sccampos** 5 months, 3 weeks ago

Selected Answer: B

B is the correct

upvoted 1 times

 **luxdolorosa** 8 months ago

Selected Answer: B

Of course, it's B. For example, when you perform tasks in the AWS web console, you cannot see the .tfstate file. Additionally, if we add that, you can update it to the latest state through 'tf refresh'.

upvoted 1 times

 **maze_** 9 months, 3 weeks ago

Selected Answer: B

The actual provisioned resources may drift, the state represents a snapshot of their status from *the last time* Terraform was run.

upvoted 1 times

 **hajurbau** 10 months, 1 week ago

Selected Answer: B

Question is a bit ambiguous.it says currently built, does it mean to say right after Terraform apply? Anyways feel like B is the answer

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

B. False

upvoted 1 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: B

B is the correct answer

upvoted 1 times

 **Chandru1988** 1 year, 2 months ago

B is the correct answer

upvoted 1 times

 **jvuoso** 1 year, 2 months ago

Selected Answer: B

Not necessarily

upvoted 2 times

 **connecttozee** 1 year, 3 months ago

After deployment, we can change configuration by manual. So terraform.tfstate may be different with current infra.

B is correct

upvoted 1 times

 **gargaditya** 1 year, 4 months ago

Question is a bit ambiguous-technically tfstate file reflects any deployments done VIA TERRAFORM.

Even if manual changes were made to a deployment done by terraform,a terraform refresh would actually update the tfstate file to reflect actual infrastructure.(this lets terraform know that there is a mismatch between tfstate and actual tf file).

upvoted 1 times

 **Prasad28** 1 year, 5 months ago

Answer - False, because state file doesn't matches the current infrastructure unwantedly.

upvoted 1 times

 **happychi** 1 year, 6 months ago

B IS CORRECT

upvoted 2 times

 **happychi** 1 year, 6 months ago

B TEST

upvoted 1 times

Question #2

Topic 1

One remote backend configuration always maps to a single remote workspace.

A. True

B. False

Correct Answer: A

Reference:

<https://www.terraform.io/docs/language/settings/backends/remote.html>

Workspaces

The remote backend can work with either a single remote Terraform Cloud workspace, or with multiple similarly-named remote workspaces (like `networking-dev` and `networking-prod`). The `workspaces` block of the backend configuration determines which mode it uses:

- To use a single remote Terraform Cloud workspace, set `workspaces.name` to the remote workspace's full name (like `networking`).

Community vote distribution

B (85%)

A (15%)

 **Zam88** Highly Voted 2 years ago

Selected Answer: B

correct B

upvoted 9 times

 **Spandrop** Highly Voted 7 months ago

Selected Answer: B

1 workspace -> 1 backend

1 backend -> multiple workspaces

upvoted 5 times

✉  **hrajkuma** Most Recent 3 days, 15 hours ago

vote for B. False

upvoted 1 times

✉  **090200f** 1 week ago

The remote backend can work with either a single remote HCP Terraform workspace, or with multiple similarly-named remote workspaces, so is correct answer

upvoted 1 times

✉  **liuyomz** 2 months, 3 weeks ago

Selected Answer: B

B. prefix switching

upvoted 1 times

✉  **6957dbd** 4 months, 1 week ago

Selected Answer: B

The prefix switch is key here:

prefix - (Optional) A prefix used in the names of one or more remote workspaces, all of which can be used with this configuration. The full workspace names are used in Terraform Cloud, and the short names (minus the prefix) are used on the command line for Terraform CLI workspaces. If omitted, only the default workspace can be used. This option conflicts with name.

To use multiple remote workspaces, set workspaces.prefix to a prefix used in all of the desired remote workspace names. For example, set pr = "networking-" to use Terraform cloud workspaces with names like networking-dev and networking-prod. This is helpful when mapping multiple Terraform CLI workspaces used in a single Terraform configuration to multiple Terraform Cloud workspaces.

The backend configuration requires either name or prefix. Omitting both or setting both results in a configuration error.

upvoted 3 times

✉  **TigerInTheCloud** 6 months, 4 weeks ago

Selected Answer: B

<https://developer.hashicorp.com/terraform/language/settings/backends/remote>

"To use multiple remote workspaces, set workspaces.prefix to a prefix used in all of the desired remote workspace names. For example, set prefix = "networking-" to use Terraform cloud workspaces with names like networking-dev and networking-prod. This is helpful when mapping multiple Terraform CLI workspaces used in a single Terraform configuration to multiple Terraform Cloud workspaces."

upvoted 4 times

✉  **luxdolorosa** 8 months ago

Selected Answer: B

I am working with GitLab, and I can create multiple instances within a single backend space

upvoted 3 times

✉  **NKSINGH** 8 months, 1 week ago

Correct B

upvoted 1 times

✉  **esuarve** 9 months, 2 weeks ago

Selected Answer: B

Your explanation actually states that a remote backend can work with multiple similarly-named remote workspaces (like networking-dev and networking-prod)

The answer A would not allow this

upvoted 2 times

✉  **arnabsinha4u** 10 months ago

Selected Answer: B

correct B

<https://developer.hashicorp.com/terraform/language/settings/backends/remote#workspaces>

upvoted 1 times

✉  **ledjo** 10 months, 3 weeks ago

B.

A remote backend configuration is used to define where the Terraform state is stored, and it can be shared by multiple workspaces, each of which maintains its own separate state.

upvoted 2 times

 **paromanu007** 11 months ago

The remote backend can work with either a single remote Terraform Cloud workspace, or with multiple similarly-named remote workspaces (like networking-dev and networking-prod). The workspaces block of the backend configuration determines which mode it uses:

Correct B

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

B. False

upvoted 1 times

 **Bere** 11 months, 3 weeks ago

Selected Answer: B

1) Example using workspaces.name

This example demonstrates the use of a single remote Terraform Cloud workspace by specifying the workspace name.
terraform {

```
...  
workspaces {  
  name = "networking-prod"  
}  
}  
}
```

In this configuration, all the Terraform commands will be applied to the networking-prod workspace in Terraform Cloud.

2) Example using workspaces.prefix

This example demonstrates the use of multiple remote Terraform Cloud workspaces that share a common prefix.

terraform {

```
...  
workspaces {  
  prefix = "networking-"  
}  
}  
}
```

In this configuration, Terraform commands will apply to any workspace that starts with networking-, such as networking-dev, networking-prod, networking-test, etc. The specific workspace that the commands apply to is determined by the currently selected Terraform CLI workspace. You can switch between workspaces using the terraform workspace select command.

upvoted 2 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: B

the correct answer B

upvoted 1 times

 **ska_torrent_430** 1 year, 1 month ago

Selected Answer: B

Correct B

upvoted 1 times

Question #3

Topic 1

How is the Terraform remote backend different than other state backends such as S3, Consul, etc.?

- A. It can execute Terraform runs on dedicated infrastructure on premises or in Terraform Cloud
- B. It doesn't show the output of a terraform apply locally
- C. It is only available to paying customers
- D. All of the above

Correct Answer: A

If you and your team are using Terraform to manage meaningful infrastructure, we recommend using the remote backend with Terraform Cloud or Terraform Enterprise.

Reference:

<https://www.terraform.io/docs/language/settings/backends/index.html>

Community vote distribution

A (100%)

 **softarts** Highly Voted 2 years, 2 months ago

I take A

upvoted 9 times

 **hrajkuma** Most Recent 3 days, 15 hours ago

vote for A

upvoted 1 times

 **pangchn** 2 months ago

Selected Answer: A

The remote backend is unique among all other Terraform backends because it can both store state snapshots and execute operations for HC Terraform's CLI-driven run workflow. It used to be called an "enhanced" backend ref

<https://developer.hashicorp.com/terraform/language/settings/backends/remote>

upvoted 2 times

 **liuyomz** 2 months, 3 weeks ago

Selected Answer: A

A for sure

upvoted 1 times

 **enook** 6 months, 2 weeks ago

Selected Answer: A

I think A is correct.

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

A. It can execute Terraform runs on dedicated infrastructure on premises or in Terraform Cloud Most Voted

upvoted 1 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: A

i take A

upvoted 1 times

 **wsyh** 1 year, 2 months ago

Selected Answer: A

Vote A

upvoted 1 times

 **meriadoc71** 1 year, 2 months ago

From ChatGPT seems that the D is the correct answer.

upvoted 1 times

 **joyboy23** 1 year ago

Well, answer this: ChatGPT is 100% trusted source. a. False b. False

upvoted 11 times

 **erictaquera** 1 year, 5 months ago

Selected Answer: A

B and C are incorrect. Correct A
upvoted 1 times

 **Atta33** 1 year, 7 months ago

A is correct

upvoted 2 times

 **stalos** 1 year, 7 months ago

Selected Answer: A

It is A

upvoted 1 times

 **Anderson01** 1 year, 8 months ago

why not C? should be multiple choices, right?

upvoted 1 times

 **30th** 2 weeks, 5 days ago

There is a limited free tier.

upvoted 1 times

 **Atta33** 1 year, 9 months ago

Selected Answer: A

A is correct

upvoted 1 times

 **Janan** 1 year, 11 months ago

Selected Answer: A

A is the answer

upvoted 2 times

 **Eltooth** 2 years ago

Selected Answer: A

A is correct answer.

upvoted 3 times

 **Zam88** 2 years ago

Backends define where Terraform's state snapshots are stored.

A given Terraform configuration can either specify a backend, integrate with Terraform Cloud, or do neither and default to storing state locally.

correct A

upvoted 4 times

Question #4

Topic 1

What is the workflow for deploying new infrastructure with Terraform?

- A. terraform plan to import the current infrastructure to the state file, make code changes, and terraform apply to update the infrastructure.
- B. Write a Terraform configuration, run terraform show to view proposed changes, and terraform apply to create new infrastructure.
- C. terraform import to import the current infrastructure to the state file, make code changes, and terraform apply to update the infrastructure.
- D. Write a Terraform configuration, run terraform init, run terraform plan to view planned infrastructure changes, and terraform apply to create new infrastructure.

Correct Answer: D

Community vote distribution

D (100%)

softarts Highly Voted 2 years, 2 months ago

it is D
upvoted 16 times

hrajkuma Most Recent 3 days, 15 hours ago

vote for D
upvoted 1 times

kambarami 2 months ago

Its DDDD yes!
upvoted 1 times

enook 6 months, 2 weeks ago

Selected Answer: D

it's DDDDDDDDD
upvoted 1 times

anon1234 7 months, 3 weeks ago

Selected Answer: D

DDDDDDDDDD
upvoted 1 times

Jayanth 11 months, 3 weeks ago

D. Write a Terraform configuration, run terraform init, run terraform plan to view planned infrastructure changes, and terraform apply to create new infrastructure.
upvoted 1 times

tfdestroy 11 months, 3 weeks ago

D = Write a Terraform configuration, run terraform init, run terraform plan to view planned infrastructure changes, and terraform apply to create new infrastructure.
upvoted 1 times

Busi57 11 months, 4 weeks ago

Selected Answer: D

D write init "plan" apply
upvoted 1 times

matthieu_chappaz 12 months ago

Selected Answer: D

D is the way
upvoted 1 times

wsyh 1 year, 2 months ago

Selected Answer: D

Vote D
upvoted 2 times

Chandru1988 1 year, 2 months ago

Definitely its D
upvoted 1 times

Power123 1 year, 3 months ago

D is correct
upvoted 1 times

Prasad28 1 year, 5 months ago

D is Correct
upvoted 1 times

👤 **InformationOverload** 1 year, 6 months ago

Selected Answer: D

yes, D

upvoted 1 times

👤 **denismaggior8** 1 year, 7 months ago

Selected Answer: D

D is the answer

upvoted 1 times

👤 **stalos** 1 year, 7 months ago

Selected Answer: D

It is D

upvoted 1 times

👤 **geekneek** 1 year, 11 months ago

D, no doubt about it.

upvoted 3 times

Question #5

Topic 1

A provider configuration block is required in every Terraform configuration.

Example:

```
provider "provider_name" {  
    ...  
}
```

A. True

B. False

Correct Answer: A

Reference:

<https://github.com/hashicorp/terraform/issues/17928>

Community vote distribution

B (68%)

A (32%)

👤 **pabrojo**  1 year, 11 months ago

Selected Answer: B

It's B. From the official documentation:

Unlike many other objects in the Terraform language, a provider block may be omitted if its contents would otherwise be empty. Terraform assumes an empty default configuration for any provider that is not explicitly configured.

<https://www.terraform.io/language/providers/configuration>

upvoted 43 times

 **gargaditya** 1 year, 4 months ago

Still need atleast 1 provider in the terraform configuration-If I am deploying to Azure, I can skip the AWS provider. But without the provider block containing details like authentication, how will the deployment actually happen?
upvoted 6 times

 **marcin3dm** 1 year, 1 month ago

AZ CLI can be used as an authentication source.
upvoted 1 times

 **Sekir** 8 months, 2 weeks ago

This question is basically asking "is this formatting correct", in which case it is.
upvoted 1 times

 **softarts** Highly Voted  2 years, 2 months ago

Selected Answer: A

vote A

upvoted 36 times

 **hrajkuma** Most Recent  3 days, 15 hours ago

vote for B. False
A provider configuration block is not required in every Terraform configuration.
Provider configuration blocks are only required when you are using a particular provider to interact with a specific type of infrastructure resource.
If your configuration does not interact with any resources provided by external providers, then you do not need to include a provider configuration block. :)
upvoted 1 times

 **brundabanm** 1 month ago

Selected Answer: B

Unlike many other objects in the Terraform language, a provider block may be omitted if its contents would otherwise be empty. Terraform assumes an empty default configuration for any provider that is not explicitly configured.

<https://developer.hashicorp.com/terraform/language/providers/configuration>

upvoted 1 times

 **Molly1994** 1 month, 1 week ago

the answer is false. Terraform requires provider. but it does not require specifically to define a provider block {}. Terraform could use the default providers. so the answer is B false.
upvoted 1 times

 **liuyomz** 2 months, 3 weeks ago

Selected Answer: B

B. i got it wrong but its on docs
upvoted 2 times

 **Bedmed** 3 months, 2 weeks ago

Each Terraform module must declare which providers it requires, so that Terraform can install and use them. Provider requirements are declared in a required_providers block.

A provider requirement consists of a local name, a source location, and a version constraint:

upvoted 1 times

✉️  **vibzr2023** 3 months, 2 weeks ago

If your Terraform configuration only includes resources from a single provider and doesn't require any special configuration for that provider, you might not need an explicit provider block. Terraform can automatically download and use the latest version of the required provider based on resource types used.

```
# Example without an explicit provider block
resource "aws_instance" "example" {
  ami = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}
```

In this example, Terraform can infer that the AWS provider is needed because of the aws_instance resource. It will use the default configuration for the AWS provider, assuming credentials and region are configured through environment variables or shared credentials files.

upvoted 2 times

✉️  **vibzr2023** 3 months, 2 weeks ago

Saying the Selected Answer:B

When You Need an Explicit Provider Block:

Example: In scenarios where you need to configure specific settings for a provider, like credentials, region, or aliases for managing resources in multiple regions or with different accounts, you will need an explicit provider block.

```
# Example with an explicit provider block
provider "aws" {
  region = "us-west-2"
  access_key = "my-access-key"
  secret_key = "my-secret-key"
}
```

```
resource "aws_instance" "example" {
  ami = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}
```

In this example, the AWS provider is explicitly configured with a specific region and credentials. This is necessary if you're not relying on the default credential chain or if you want to set parameters that differ from the defaults.

upvoted 2 times

✉️  **6957dbd** 4 months, 1 week ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/language/providers/configuration>

Additionally, all Terraform configurations must declare which providers they require so that Terraform can install and use them. The Provider Requirements page documents how to declare providers so Terraform can install them.

upvoted 1 times

✉️  **AWSCurt** 5 months, 1 week ago

Selected Answer: B

False. A provider configuration block is not required in every Terraform configuration. Provider configuration blocks are only required when you are using a particular provider to interact with a specific type of infrastructure resource. If your configuration does not interact with any resources provided by external providers, then you do not need to include a provider configuration block.

upvoted 2 times

✉️  **MukeshRattan** 5 months, 3 weeks ago

Selected Answer: B

False.

A provider configuration block is not required in every Terraform configuration. It depends on the specifics of your configuration and the resources you are managing.

In Terraform, provider configuration blocks are used to specify the details of the infrastructure provider you want to use, such as AWS, Azure, Google Cloud, etc. If you are managing resources that don't require a specific provider, or if your configuration relies on provider-agnostic resources, you may not need a provider configuration block.

upvoted 1 times

✉️  **vipulchoubisa** 6 months, 1 week ago

if example is given as provider "provider_name" [...] then it should be A answer else B. I will go with A

upvoted 1 times

✉️  **samimshaikh** 6 months, 2 weeks ago

Selected Answer: B

False. A provider configuration block is not required in every Terraform configuration. It is only required when you are using a Terraform provider to interact with a specific infrastructure platform or service.

A provider configuration block typically includes details such as the provider's name, version, and any required authentication or connection information. If you're not using any provider in your Terraform configuration, you may not need a provider configuration block.

Here's an example of a provider configuration block for AWS:

```
provider "aws" {  
region = "us-west-2"  
access_key = "your-access-key"  
secret_key = "your-secret-key"  
}
```

This block specifies the AWS provider, sets the region, and provides access and secret keys for authentication. If you're not working with AWS or any other provider, you can have a Terraform configuration without a provider block.

upvoted 1 times

✉️  **TigerInTheCloud** 6 months, 4 weeks ago

Selected Answer: B

there is a simple valid configuration without the provider BLOCK.

```
$ cat main.tf  
data "aws_region" "current" {}  
  
output "region_name" {  
value = data.aws_region.current.name  
}  
  
upvoted 1 times
```

✉️  **Spandrop** 7 months ago

Selected Answer: B

A Terraform configuration is a complete document in the Terraform language that tells Terraform how to manage a given collection of infrastructure. A configuration can consist of multiple files and directories.

<https://developer.hashicorp.com/terraform/language>

You don't need 1 provider block to EVERY terraform configuration, you must have at least 1, but not in every like the question mention

upvoted 1 times

Question #6

Topic 1

You run a local-exec provisioner in a null resource called null_resource.run_script and realize that you need to rerun the script.

Which of the following commands would you use first?

- A. `terraform taint null_resource.run_script`
- B. `terraform apply -target=null_resource.run_script`
- C. `terraform validate null_resource.run_script`
- D. `terraform plan -target=null_resource.run_script`

Correct Answer: A

Community vote distribution

A (72%)

B (21%)

7%

✉️  **alirasouli** Highly Voted 1 year, 8 months ago

As discussed, the 'taint' command used to be the right choice; however, it is deprecated. The right answer is:
`terraform apply -replace="null_resource.run_script"`

Reference:

<https://developer.hashicorp.com/terraform/cli/commands/taint>

upvoted 31 times

✉  **Sproket**  2 years ago

Selected Answer: A

You are all correct that taint has been deprecated and replaced with -replace. But neither D nor any other option here uses the -replace command. Therefore option A is the only valid option given these choices.

upvoted 17 times

✉  **Pietjeplukgeluk** 1 year, 5 months ago

Not so sure, read Arrash his comment below. If a provisioner fails, it would be marked as tainted by default without any user interaction required. This only leaves the apply as a required step....

upvoted 3 times

✉  **hrajkuma**  3 days, 15 hours ago

it is option -> A for sure

upvoted 1 times

✉  **090200f** 1 week ago

Warning: This command is deprecated. For Terraform v0.15.2 and later, we recommend using the -replace option with terraform apply instead So { \$ terraform apply -replace="aws_instance.example[0]" } but there is no like this apply -replace option in the mentioned options so answer A : terraform taint

upvoted 1 times

✉  **enklau** 1 week ago

thats right

upvoted 1 times

✉  **gofavad926** 9 months, 2 weeks ago

Selected Answer: A

A. However taint command is deprecated now and we should use terraform apply -destroy

upvoted 2 times

✉  **ledjo** 10 months, 3 weeks ago

Selected Answer: A

terraform apply -replace is missing here, therefore the only valid answer, even if deprecated as a command, is the first one A.

upvoted 2 times

✉  **BaburTurk** 10 months, 3 weeks ago

Selected Answer: B

B. terraform apply -target=null_resource.run_script

Running the terraform apply -target=null_resource.run_script command will specifically target the null_resource.run_script resource and execute its provisioner again. This is useful when you want to rerun the local-exec provisioner without affecting other resources.

upvoted 3 times

✉  **Jayanth** 11 months, 3 weeks ago

A. terraform taint null_resource.run_script

upvoted 1 times

✉️  **Bere** 11 months, 3 weeks ago

Selected Answer: A

1) Create main.tf:
terraform {
 required_version = ">= 0.13"
}

```
resource "null_resource" "run_script" {  
  provisioner "local-exec" {  
    command = "echo 'Hello, Terraform!' > example.txt"  
  }  
}
```

- 2) terraform init
- 3) terraform apply
- 4) Modify the content of the example.txt file manually or delete it.
- 5) terraform taint null_resource.run_script
- 6) terraform apply

The local-exec provisioner should now run again, and the example.txt file will be recreated with the content specified in the command of the local-exec provisioner block.

As described here:

<https://developer.hashicorp.com/terraform/cli/commands/taint>

This command is deprecated. For Terraform v0.15.2 and later, we recommend using the -replace option with terraform apply instead

But there is no option that uses the -replace command, so option A as described in my step 5 is the only valid option.

upvoted 2 times

✉️  **Busi57** 11 months, 4 weeks ago

Selected Answer: A

I think A

upvoted 1 times

✉️  **nebulabc** 1 year ago

To rerun the script defined in the local-exec provisioner of a null resource called null_resource.run_script, you can use the terraform apply command with the -target flag to specifically target the null resource.

This command instructs Terraform to only apply changes to the specified target resource, which in this case is the null_resource.run_script. It re-run the local-exec provisioner associated with that null resource.

upvoted 1 times

✉️  **nebulabc** 1 year ago

So, B is correct.

upvoted 1 times

✉️  **Ni33** 1 year, 2 months ago

Selected Answer: A

A is correct

upvoted 2 times

✉️  **FarziWaliMarzi** 1 year, 3 months ago

Selected Answer: A

I think point to note is "you realize that you need to rerun the script". It is NOT talking about failure. So marking it taint is the only right option, that on next run, it can be deleted and recreated.

upvoted 2 times

✉️  **aanataliya** 10 months, 2 weeks ago

you dont need to taint to re run script in local-exec provisioner with null resource.
reference: <https://jhooq.com/terraform-null-resource/>

upvoted 1 times

✉️  **Power123** 1 year, 3 months ago

terraform apply -replace="null_resource.run_script"

Ans: A

upvoted 1 times

 **Nunyabiznes** 1 year, 3 months ago

The correct answer is A. `terraform taint null_resource.run_script`. This command marks the null resource as tainted, which means that Terraform considers the resource to be out-of-date and will recreate it during the next `terraform apply` run. When Terraform recreates the null resource, it also reruns the local-exec provisioner.

Option B, `terraform apply -target=null_resource.run_script`, would work, but it is overkill because it would apply all the resources in the configuration, not just the null resource with the local-exec provisioner.

Option C, `terraform validate null_resource.run_script`, only checks the syntax of the configuration, and does not affect the state of the resources.

Option D, `terraform plan -target=null_resource.run_script`, generates a plan for applying changes to the configuration, but does not apply those changes, so it would not rerun the local-exec provisioner.

upvoted 3 times

 **gargaditya** 1 year, 4 months ago

(1/3)

Answer should be B.

The first thing to realise is that the question is testing failure behaviour of provisioners.

Hence the need of re-running the script.

upvoted 1 times

 **gargaditya** 1 year, 4 months ago

(2/3)=

This is my understanding:

By default, provisioners that fail will also cause the Terraform `apply` itself to fail. The '`on_failure`' setting can be used to change this. The allowed values are:

- `continue` - Ignore the error and continue with creation or destruction.

- This also means if there are multiple commands clubbed [using & in Windows or ; in Linux] (in the inline or command argument) or multiple provisioners following, they will execute.

- Also, other following terraform resources(based on computed dependencies) will continue to get created/destroyed.

- `fail` - Raise an error and stop applying OR destroying (the default behaviour).

> If this is a creation provisioner, taint the resource.

> Destroy provisioners are run before the resource is destroyed. If they fail, Terraform will error and rerun the provisioners again on the next `terraform apply`. Due to this behavior, care should be taken for destroy provisioners to be safe to run multiple times.

upvoted 1 times

 **gargaditya** 1 year, 4 months ago

(3/3)

- This also means if there are multiple commands clubbed [using & in Windows or ; in Linux] (in the inline or command argument) or multiple provisioners following, they will NOT execute.

- Also, other following terraform resources(based on computed dependencies) will NOT get created/destroyed

==

With no specific input on '`on_failure`', the default action is '`fail`'-ie either taint the resource(creation time provisioner) or AUTOMATIC re-run.

Question #7

Topic 1

Which provisioner invokes a process on the resource created by Terraform?

- A. remote-exec
- B. null-exec
- C. local-exec
- D. file

Correct Answer: A

The remote-exec provisioner invokes a script on a remote resource after it is created.

Reference:

<https://www.terraform.io/docs/language/resources/provisioners/remote-exec.html>

Community vote distribution

A (100%)

 **Cloud9er**  2 years, 2 months ago

No. Answer is remote-exec. because...The local-exec provisioner invokes a local executable after a resource is created. This invokes a process on the machine running Terraform, not on the resource. (this is from the terraform web site)
upvoted 22 times

 **Eltooth**  2 years ago

Selected Answer: A

A is correct answer : remote-exec.

"The local-exec provisioner invokes a local executable after a resource is created. This invokes a process on the machine running Terraform, not on the resource." <https://www.terraform.io/language/resources/provisioners/local-exec>

"The remote-exec provisioner invokes a script on a remote resource after it is created."
<https://www.terraform.io/language/resources/provisioners/remote-exec>

upvoted 5 times

 **090200f**  1 week ago

A is correct ans : "process on the resource" is key point here. so remote-exec. The local-exec provisioner invokes a local executable after a resource is created. This invokes a process on the machine running Terraform, not on the resource.

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

A. remote-exec

upvoted 1 times

 **Bere** 11 months, 3 weeks ago

Selected Answer: A

The remote-exec provisioner invokes a script on a remote resource created by Terraform. It connects to the resource using SSH or WinRM and runs the provided inline or script commands. Here's an example of how you might use it:

```
resource "aws_instance" "example" {  
  ami = "ami-0c94855ba95c574c8"  
  instance_type = "t2.micro"  
  
  provisioner "remote-exec" {  
    inline = [  
      "echo Hello, World! > /home/ubuntu/hello",  
      "chmod +x /home/ubuntu/hello",  
    ]  
  }  
}
```

There's no null-exec provisioner in Terraform.

The local-exec provisioner invokes a local executable after a resource is created. It runs on the machine where Terraform is being executed.

The file provisioner is used to copy files or directories from the machine executing Terraform to the newly created resource.

upvoted 3 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: A

remote-exec -> A

upvoted 1 times

 **Mandeeps468** 1 year, 1 month ago

A - correct answer : remote-exec

upvoted 1 times

 **wsyh** 1 year, 1 month ago

Selected Answer: A

Vote A. local-exec provisioner is execute on the local machine.

upvoted 1 times

 **Chandru1988** 1 year, 2 months ago

Definitely Remote-exec

upvoted 1 times

 **Power123** 1 year, 3 months ago

A is correct

upvoted 1 times

 **Mal_8** 1 year, 4 months ago

Selected Answer: A

A is the correct answer : remote-exec

upvoted 3 times

 **ACE_ASPIRE** 1 year, 10 months ago

This question was in my exam.

upvoted 4 times

 **Zam88** 2 years ago

The remote-exec provisioner invokes a script on a remote resource after it is created. This can be used to run a configuration management to bootstrap into a cluster, etc. To invoke a local process, see the local-exec provisioner instead. The remote-exec provisioner requires a connection and supports both ssh and winrm.

remote-exec correct

upvoted 1 times

 **subhala** 2 years ago

I think it is remote-exec, the question is asking about the resource created by terraform. local-exec runs on the machine where we run terraform.

upvoted 1 times

 **softarts** 2 years, 2 months ago

I take A remote-exec.

upvoted 3 times

 **Eniras** 2 years, 2 months ago

The local-exec provisioner invokes a local executable after a resource is created. This invokes a process on the machine running Terraform, not on the resource. See the remote-exec provisioner to run commands on the resource.

<https://www.terraform.io/language/resources/provisioners/local-exec>

upvoted 3 times

 **calebvar** 2 years, 2 months ago

Isn't local exec also correct?

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

Eniras explained well about the Local Exec,

upvoted 1 times

Question #8

Topic 1

Which of the following is not true of Terraform providers?

- A. Providers can be written by individuals
- B. Providers can be maintained by a community of users

- C. Some providers are maintained by HashiCorp
- D. Major cloud vendors and non-cloud vendors can write, maintain, or collaborate on Terraform providers
- E. None of the above

Correct Answer: D

Reference:

https://jayendrapatil.com/terraform-cheat-sheet/#Terraform_Read_and_write_configuration

Community vote distribution

E (95%)

5%

 **icedog** Highly Voted 2 years, 2 months ago

Selected Answer: E

E. is the correct answer.

It's obvious that D. is incorrect, of course cloud and non-cloud vendors can collaborate on creating providers just like anyone else upvoted 37 times

 **nharaz** Highly Voted 1 year, 5 months ago

Selected Answer: E

E. None of the above

All of the statements are true of Terraform providers.

A. Providers can be written by individuals - Any person or organization can develop and distribute a Terraform provider, allowing them to expand Terraform's capabilities to manage resources that it previously could not.

B. Providers can be maintained by a community of users - Many Terraform providers are open source projects, and the development and maintenance of these providers can be collaborative efforts between multiple individuals and organizations.

C. Some providers are maintained by HashiCorp - HashiCorp, the creators of Terraform, maintain a number of official providers that cover popular infrastructure providers such as AWS, Google Cloud, and Microsoft Azure.

D. Major cloud vendors and non-cloud vendors can write, maintain, or collaborate on Terraform providers - Providers can be developed and maintained by cloud vendors, non-cloud vendors, or a combination of both, to expand Terraform's capabilities and support for different types of infrastructure.

upvoted 13 times

 **hrajkuma** Most Recent 3 days, 14 hours ago

E is the right answer

upvoted 1 times

 **Narendra_Sharma** 1 month, 3 weeks ago

Selected Answer: E

all statement are true

upvoted 1 times

 **pangchn** 2 months ago

Selected Answer: E

<https://developer.hashicorp.com/terraform/registry/providers>

Official

Partner

Community

upvoted 1 times

 **galvarado89** 5 months, 1 week ago

Selected Answer: D

cloud and non-cloud vendors can collaborate on creating providers just like anyone else

upvoted 1 times

 **samimshaikh** 6 months, 2 weeks ago

Selected Answer: E

E. None of the above

All of the statements (A, B, C, and D) are true of Terraform providers:

A. Providers can be written by individuals: True. Terraform allows individuals to create custom providers for their specific needs.

B. Providers can be maintained by a community of users: True. Terraform has a community-driven model, and many providers are developed and maintained by the community.

C. Some providers are maintained by HashiCorp: True. HashiCorp, the company behind Terraform, maintains and supports certain official providers.

D. Major cloud vendors and non-cloud vendors can write, maintain, or collaborate on Terraform providers: True. Terraform is designed to work with a variety of providers, including major cloud providers, on-premises solutions, and other services. Providers can be developed and maintained by the vendors themselves or in collaboration with the community.

upvoted 1 times

 **guicane** 7 months, 3 weeks ago

Selected Answer: E

it's obviously E

upvoted 1 times

 **qs5dq64s5dq5** 8 months, 3 weeks ago

Selected Answer: E

D is obviously true

upvoted 1 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: E

E. None of the above

upvoted 1 times

 **rajat06** 9 months, 3 weeks ago

E is the correct None of the above

upvoted 1 times

 **ledjo** 10 months, 3 weeks ago

Selected Answer: E

E. is the correct answer.

upvoted 1 times

 **Ashutosh_96** 11 months, 3 weeks ago

E, is correct

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

E. None of the above

upvoted 1 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: E

all true -> E

upvoted 1 times

 **Shane_C** 1 year ago

Selected Answer: E

E is clearly the correct answer

upvoted 1 times

 **abhi6199** 1 year ago

Answer is E.

upvoted 1 times

What command does Terraform require the first time you run it within a configuration directory?

- A. terraform import
- B. terraform init
- C. terraform plan
- D. terraform workspace

Correct Answer: B

terraform init command is used to initialize a working directory containing Terraform configuration files.

Reference:

<https://www.terraform.io/docs/cli/commands/init.html>

Community vote distribution

B (100%)

 **softarts** Highly Voted 2 years, 2 months ago

it is B

upvoted 8 times

 **hrajkuma** Most Recent 3 days, 14 hours ago

B is the answer, terraform init

upvoted 1 times

 **090200f** 1 week ago

B. terraform init is correct answer it initializes all resource config need to current directory (path)

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

Question #10

Topic 1

You have deployed a new webapp with a public IP address on a cloud provider. However, you did not create any outputs for your code.

What is the best method to quickly find the IP address of the resource you deployed?

- A. Run terraform output ip_address to view the result
- B. In a new folder, use the terraform_remote_state data source to load in the state file, then write an output for each resource that you find the state file
- C. Run terraform state list to find the name of the resource, then terraform state show to find the attributes including public IP address
- D. Run terraform destroy then terraform apply and look for the IP address in stdout

Correct Answer: A

Reference:

<https://www.terraform.io/docs/cli/commands/output.html>

Community vote distribution

C (100%)

 **coco10k** Highly Voted 2 years, 2 months ago

Selected Answer: C

You can find the info in the state.

not A because you don't have to outputs defined.

upvoted 29 times

 **hrajkuma** Most Recent 3 days, 14 hours ago

vote for option C

upvoted 1 times

 **brundabanm** 1 month ago

C is the correct answer.

upvoted 1 times

 **samimshaikh** 6 months, 2 weeks ago

Selected Answer: C

C. Run terraform state list to find the name of the resource, then terraform state show to find the attributes, including the public IP address.

Explanation:

Not Option A (terraform output ip_address) assumes that there is an output variable named ip_address in your Terraform configuration, but since you mentioned that you did not create any outputs, this option may not be applicable.

upvoted 2 times

 **Prat8** 7 months, 3 weeks ago

Is it C or A ?

upvoted 1 times

 **luke404** 8 months, 2 weeks ago

Selected Answer: C

The question does not talk about any output defined so you can't expect one

upvoted 1 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: C

C for sure

upvoted 1 times

 **otakuinside** 9 months, 3 weeks ago

Selected Answer: C

A cannot be. You don't have any output declared, so there is no default output or something like that where you could find the ip address. On other hand, terraform state list, and then show specific, contains the state and this includes the ip address

upvoted 2 times

 **grogudev** 9 months, 3 weeks ago

Selected Answer: C

C I choose too

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

C. Run terraform state list to find the name of the resource, then terraform state show to find the attributes including public IP address
upvoted 1 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: C

C i choose

upvoted 1 times

 **cdechery** 1 year ago

Selected Answer: C

If there is no output defined for the resource, the "output" command will not show anything. Correct is C.

upvoted 2 times

 **Shane_C** 1 year ago

Selected Answer: C

C is the correct answer.

It can't be A as the output isn't defined

upvoted 1 times

 **SIAMIANJI** 1 year, 1 month ago

Selected Answer: C

C is correct,

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: C

C is the correct answer

upvoted 1 times

 **connecttozee** 1 year, 3 months ago

100% is C

upvoted 2 times

 **camps** 1 year, 3 months ago

Selected Answer: C

C. Run terraform state list to find the name of the resource, then terraform state show to find the attributes including public IP address is the method to quickly find the IP address of the resource you deployed.

terraform state list will show you the name of all the resources currently tracked in your Terraform state. You can then use terraform state show followed by the resource name to see all the details of the resource, including any attributes like the public IP address.

upvoted 2 times

Question #11

Topic 1

Which of the following is not a key principle of infrastructure as code?

- A. Versioned infrastructure
- B. Golden images
- C. Idempotence
- D. Self-describing infrastructure

Correct Answer: ABD

Reference:

<https://docs.microsoft.com/en-us/azure/devops/learn/what-is-infrastructure-as-code#:~:text=Idempotence%20is%20a%20principle%20of,of%20the%20environment's%20starting%20state>

Community vote distribution

B (92%)

8%

 **ItaloVinodi** Highly Voted 2 years, 2 months ago

Sorry but here is asking which is NOT a key principle. I think the answer is B, golden Image

upvoted 28 times

 **alexik96** Highly Voted 1 year, 9 months ago

B is correct. Even if you're not sure, you can eliminate the others

A - is a principle (think of the case when you store the state in an S3 bucket and it gets versioned)

B - is NOT a principle (where in terraform documentation is mentioned anything related to images)

C - it is a principle by definition of what terraform does

D - is a principle as you use declarative language -> infrastructure describes itself

upvoted 11 times

 **starkonbullet** Most Recent 1 month, 4 weeks ago

Selected Answer: B

The other 3 options can be associated with the principle.

upvoted 1 times

 **SilentH** 3 months ago

Selected Answer: B

I vote B because not only is a "golden image" not a key principle but, I would argue, in conflict with the idea behind IaC.

upvoted 1 times

 **vibzr2023** 3 months, 2 weeks ago

Answer:B

B. Golden Images: While golden images (pre-configured images for server deployments) are used in infrastructure management, they are not considered a key principle of IaC. IaC focuses on defining infrastructure through code that can be versioned, reused, and managed as part of application development workflows. Relying solely on golden images does not fully embrace the dynamism and flexibility offered by IaC practices, as it leans more towards a traditional, immutable infrastructure approach.

upvoted 2 times

 **samimshaikh** 6 months, 2 weeks ago

B. Golden images: While golden images are a common concept in traditional infrastructure management, they are not a key principle of infrastructure as code. IaC focuses on defining and managing infrastructure through code rather than relying on pre-configured images.

upvoted 2 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: B

Agree with B

upvoted 1 times

 **Badii** 10 months, 1 week ago

Selected Answer: B

Golden images are not a key principle of infrastructure as code (IaC). Instead, golden images are a concept related to traditional server provisioning and management.

upvoted 1 times

 **Jayanth** 11 months, 3 weeks ago

B. Golden images

upvoted 1 times

 **Bere** 11 months, 3 weeks ago

Selected Answer: B

Golden images are not a key principle of infrastructure as code (IaC). Golden images refer to a pre-configured and standardized system image that contains all the necessary software and settings to run a particular application or service. While they are commonly used in traditional server management and virtualization, they are not inherently related to infrastructure as code.

upvoted 3 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: B

B Golden image

upvoted 1 times

 **foreverlearner** 1 year ago

Selected Answer: B

Golden image is not part of IaC.

For more info see <https://developer.hashicorp.com/well-architected-framework/operational-excellence/operational-excellence-tao>

upvoted 1 times

 **Shane_C** 1 year ago

Selected Answer: B

B is the only answer here that is not a key principle

upvoted 1 times

 **connecttozee** 1 year, 3 months ago

agree B

upvoted 1 times

 **Power123** 1 year, 3 months ago

B is correct - Golden image

upvoted 1 times

 **Nunyabiznes** 1 year, 3 months ago

Only one option can be selected as the question is asking for a principle that is not a key principle of infrastructure as code. Based on that, the answer would be:

B. Golden images

upvoted 1 times

 **SilentMilli** 1 year, 4 months ago

Selected Answer: B

Golden images refer to pre-configured, fully patched system images that are used to quickly deploy new servers. While golden images can be useful for speeding up server provisioning, they are not a key principle of infrastructure as code.

upvoted 3 times

Question #12

Topic 1

Terraform variables and outputs that set the "description" argument will store that description in the state file.

A. True

B. False

Correct Answer: A

Community vote distribution

B (100%)

 **SilentMilli** Highly Voted 1 year, 4 months ago

Selected Answer: B

Terraform variables and outputs that set the "description" argument are not stored in the state file. The "description" argument is used to provide a human-readable description of the variable or output, and it is intended to be used as documentation for other users of the Terraform code.

The state file is used to store the current state of the infrastructure managed by Terraform, including the values of variables and outputs. However, the "description" argument is not part of the state file, and it is not used by Terraform to manage the infrastructure. Instead, it is only used as metadata to describe the variable or output.

upvoted 16 times

 **antivrillee** Highly Voted 1 year, 10 months ago

Selected Answer: B

Answer is B. Descriptions aren't stored in the state file.

upvoted 10 times

 **starkonbullet** Most Recent 1 month, 4 weeks ago

Selected Answer: B

Validated; it would just show type, value in the terraform state.

upvoted 2 times

 **mamtak_2008** 5 months, 2 weeks ago

A. True

Terraform variables and outputs that set the "description" argument will indeed store that description in the state file. This allows for better documentation and understanding of the purpose of variables and outputs within the Terraform configuration. However, it's important to note that sensitive information should not be included in descriptions as the state file is stored in plain text and could potentially expose sensitive information if not handled properly.

upvoted 1 times

 **samimshaikh** 6 months, 2 weeks ago

Selected Answer: B

Terraform state files are intended to store the actual configuration and state of the infrastructure, focusing on information that is necessary for Terraform to understand and manage the resources. Descriptions, being informational and not affecting the infrastructure's configuration or behavior, are typically not included in the state file to keep it concise and focused on essential details for Terraform's management.

upvoted 2 times

 **guicane** 7 months, 3 weeks ago

Selected Answer: B

I don't understand how such a simple question can show the wrong answer by the website.

upvoted 1 times

✉  **luke404** 8 months, 2 weeks ago

Selected Answer: B

simply check the output of "terraform state pull" or see inside the tfstate file if using local state, there are no descriptions stored in the state
upvoted 1 times

✉  **Mimi666** 9 months, 2 weeks ago

Selected Answer: B

variables are not stored in state
upvoted 1 times

✉  **Badii** 10 months, 1 week ago

Selected Answer: B

Answer is B
upvoted 1 times

✉  **Jayanth** 11 months, 3 weeks ago

B. False

upvoted 1 times

✉  **Bere** 11 months, 3 weeks ago

Selected Answer: B

1) Create the following Terraform configuration:

```
provider "aws" {  
region = "eu-west-1"  
}
```

```
variable "instance_type" {  
description = "The type of instance to create"  
default = "t2.micro"  
}
```

```
resource "aws_instance" "example" {  
ami = "ami-096800910c1b781ba" # Replace this with an appropriate AMI ID for your region  
instance_type = var.instance_type  
}
```

```
output "instance_id" {  
description = "The ID of the created instance"  
value = aws_instance.example.id  
}
```

2) After you run terraform apply on this configuration, the state file (terraform.tfstate) will include the value of the instance_type variable ("t2.micro")
and the value of the instance_id output (which would be the ID of the created AWS instance),
but it would not include the descriptions of either the variable or the output.

upvoted 9 times

✉  **Busi57** 11 months, 4 weeks ago

Selected Answer: B

i think B

upvoted 1 times

✉  **Shane_C** 1 year ago

Selected Answer: B

B is correct

upvoted 1 times

✉  **Don5366** 1 year ago

This examtopic is getting me confused.

upvoted 5 times

✉  **SIAMIANJI** 1 year, 1 month ago

Selected Answer: B

The correct answer is B.

upvoted 2 times

 **[Removed]** 1 year, 1 month ago

Why the answers are all wrong?

upvoted 3 times

 **Power123** 1 year, 3 months ago

Correct Ans is B

upvoted 1 times

Question #13

Topic 1

What is the provider for this fictitious resource?

```
resource "aws_vpc" "main" {
    name = "test"
}
```

- A. vpc
- B. main
- C. aws
- D. test

Correct Answer: C

Reference:

<https://docs.aws.amazon.com/clouformation-cli/latest/userguide/resource-types.html>

Community vote distribution

C (100%)

 **softarts** Highly Voted 2 years, 2 months ago

C (aws)

upvoted 12 times

 **Nunyabiznes** Highly Voted 1 year, 3 months ago

In Terraform, a provider is responsible for managing resources of a particular type or a specific cloud service. In the given example, the resource type is aws_vpc which belongs to the AWS cloud platform, so the provider is specified as aws.

Therefore, the correct code snippet with the provider block would be:

```
provider "aws" {
region = "us-west-2"
}
```

```
resource "aws_vpc" "main" {
name = "test"
}
```

upvoted 9 times

 **Molly1994** Most Recent 1 month, 2 weeks ago

Resource block defines all resources from its own provider. The question is asking the provider. So answer definitely is AWS

upvoted 1 times

enook 6 months ago

Answer is C.

aws_vpc is "resource type", and main is "resource name".

upvoted 1 times

gofavad926 9 months, 2 weeks ago

Selected Answer: C

C for sure

upvoted 1 times

Jayanth 11 months, 3 weeks ago

C. aws

upvoted 1 times

Busi57 11 months, 4 weeks ago

Selected Answer: C

i choose C

upvoted 1 times

gspb 1 year, 2 months ago

Selected Answer: C

C - aws

Here, aws is the provider and aws_vpc is the resource_type

upvoted 1 times

Saifwsm 1 year, 3 months ago

C - AWS

upvoted 1 times

gargaditya 1 year, 4 months ago

Selected Answer: C

Each resource block describes one or more infrastructure objects, such as virtual networks, compute instances, or higher-level components such as DNS records.

A resource block declares a resource of a given type ("aws_instance") with a given local name ("web"). The name is used to refer to this resource from elsewhere in the same Terraform module, but has no significance outside that module's scope.

The resource type and name together serve as an identifier for a given resource and so must be unique within a module.

Terraform usually automatically determines which provider to use based on a resource type's name. (By convention, resource type names start with their provider's preferred local name.)

eg.

```
resource "aws_instance" "web" {  
  ami = "ami-a1b2c3d4"  
  instance_type = "t2.micro"  
}
```

upvoted 2 times

E_aws 1 year, 6 months ago

actually I see no error in the syntax <https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/vpc>

upvoted 1 times

E_aws 1 year, 6 months ago

sorry, misunderstood.. C is the correct answer

upvoted 1 times

terraform 1 year, 6 months ago

Selected Answer: C

C , AWS

upvoted 1 times

Smiley 1 year, 7 months ago

Selected Answer: C

C, AWS

upvoted 1 times

 **alifie** 1 year, 9 months ago

Selected Answer: C

c, aws

upvoted 1 times

 **Eltooth** 2 years ago

Selected Answer: C

C is correct answer.

upvoted 1 times

 **mk1708** 2 years ago

C (aws)

upvoted 2 times

 **cytron** 2 years, 1 month ago

C (aws)

upvoted 2 times

Question #14

Topic 1

If you manually destroy infrastructure, what is the best practice reflecting this change in Terraform?

- A. Run terraform refresh
- B. It will happen automatically
- C. Manually update the state file
- D. Run terraform import

Correct Answer: B

Community vote distribution

A (54%)

B (44%)

1%

 **tf181**  2 years, 2 months ago

How could change reflect in terraform without doing anything?

I think A is right answer. any thoughts?

upvoted 33 times

 **G_**  2 years, 2 months ago

Wouldn't it be A?

upvoted 12 times

 **callmegino** Most Recent 4 weeks ago

A is the correct answer.

Terraform says

Warning: This command is deprecated, because its default behavior is unsafe if you have misconfigured credentials for any of your providers.

You shouldn't typically need to use this command, because Terraform automatically performs the same refreshing actions as a part of creating plan in both the terraform plan and terraform apply commands.

upvoted 1 times

 **starkonbullet** 1 month, 4 weeks ago

Selected Answer: A

The command had been deprecated already. Use -refresh-only option along with terraform plan or terraform apply for the same operation.

upvoted 1 times

 **deepakpamban** 2 months, 1 week ago

terraform refresh is not deprecated. Answer is A

upvoted 1 times

 **DarrylNg** 2 months, 3 weeks ago

<https://developer.hashicorp.com/terraform/cli/commands/refresh>. A should be the answer

Need to run terraform plan or apply with refresh tag

upvoted 1 times

 **ravk2321** 4 months, 1 week ago

Selected Answer: B

"The Terraform refresh command reads the current settings from all managed remote objects and updates the Terraform state to match."

Warning: This command is deprecated because its default behaviour is unsafe if you have misconfigured credentials for any of your providers
See below for more information and recommended alternatives.

This won't modify your real remote objects, but it will modify the Terraform state.

"You shouldn't typically need to use this command, because Terraform automatically performs the same refreshing actions as a part of creating plan in both the Terraform plan and Terraform apply commands. This command is here primarily for backward compatibility, but we don't recommend using it because it provides no opportunity to review the effects of the operation before updating the state."

Link:<https://developer.hashicorp.com/terraform/cli/commands/refresh>

upvoted 2 times

 **akkam89** 4 months, 2 weeks ago

Selected Answer: B

"The terraform refresh command reads the current settings from all managed remote objects and updates the Terraform state to match."

Warning: This command is deprecated, because its default behavior is unsafe if you have misconfigured credentials for any of your providers.
See below for more information and recommended alternatives."

upvoted 2 times

 **dizzy_monkey** 4 months, 3 weeks ago

Selected Answer: A

you can do terraform refresh which is a alias for terraform apply -refresh-only -auto-approve

upvoted 2 times

 **Deva2596** 6 months ago

Selected Answer: A

upvoted 1 times

 **samimshaikh** 6 months, 2 weeks ago

Selected Answer: A

there are two ways to do this manually remove the resource using "terraform rm" from the state management and the other way is to refresh t state. Since the option given in this question is not clear enough as to run "terraform rm" or not in option C, I will go with option A is the best suited.

terraform refresh

upvoted 1 times

 **samimshaikh** 6 months, 2 weeks ago

in the context of question resource is deleted manually mean we can just refresh the state while terrafrom rm command or manually update state file updated than resources will be still there so I will go with a TERRAFORM REFRESH

upvoted 1 times

 **TigerInTheCloud** 6 months, 4 weeks ago

Selected Answer: A

A is the best answer here

terraform plan -refresh-only or terraform apply -refresh-only is best practice, but not mentioned here.

upvoted 1 times

 **Rtech75** 7 months, 3 weeks ago

The wordings on B very unclear, while A is the closest possible answer. I would choose A over B for lack option and clarity on answers given

upvoted 1 times

 8 months, 1 week ago

Due to changes in terraform versions, the answer was previously A, but is now B.

upvoted 1 times

 **Tyler2023** 8 months, 4 weeks ago

The should be B

Wherever possible, avoid using 'terraform refresh' explicitly and instead rely on Terraform's behavior of automatically refreshing existing objects as part of creating a normal plan.

If you don't want to automatically refresh, you can use the 'terraform apply -refresh-only' without the -auto-approve option

terraform refresh is deprecated

<https://developer.hashicorp.com/terraform/cli/commands/refresh>

upvoted 4 times

 **Tyler2023** 8 months, 4 weeks ago

unless, you are using old version of TF that doesn't support the refresh-only

but you shouldn't typically need to use this command, because Terraform automatically performs the same refreshing actions as a part of creating a plan in both the 'terraform plan' and 'terraform apply' commands.

upvoted 2 times

 **TigerInTheCloud** 6 months, 4 weeks ago

Read the document carefully :-)

The document mentions the reason for the deprecation and recommends alternatives.

"A" is not the best practice in the real world, but is the best answer available here.

"B" it won't automatically happen unless run some command.

upvoted 1 times

 **Badii** 9 months, 2 weeks ago

Selected Answer: A

A. Run terraform refresh

upvoted 1 times

 **rohitdabhi2026** 9 months, 2 weeks ago

Question #15

Topic 1

What is not processed when running a terraform refresh?

A. State file

B. Configuration file

C. Credentials

D. Cloud provider

Correct Answer: CD

Reference:

<https://www.terraform.io/docs/cli/commands/refresh.html>

Community vote distribution

B (98%)

2%

✉ **wangchung** Highly Voted 2 years, 1 month ago

Selected Answer: B

- A. State file - its updated during the refresh
- B. Configuration file -
- C. Credentials - required to get onto the cloud
- D. Cloud provider - required to carry out the refresh of whats in the cloud vs whats in state.

upvoted 35 times

✉ **cloudcuckooland** Highly Voted 2 years, 1 month ago

Selected Answer: B

the answer has to be B, as per docs, it's checking remote objects and update the state file, it needs to process the creds to connect, the cloud provider to check objects and the state to update.

"The terraform refresh command reads the current settings from all managed remote objects and updates the Terraform state to match."

upvoted 12 times

✉ **shubhamlonkar** Most Recent 1 week, 5 days ago

Selected Answer: B

Refresh command updates the state file but not the terraform configuration file

upvoted 1 times

✉ **Molly1994** 1 month, 2 weeks ago

Refresh command will update the remote state file to match the current configuration. So configuration file won't change anything. State file and credentials and cloud providers will associate with the refresh action

upvoted 1 times

✉ **ravk2321** 4 months, 1 week ago

The question is outdated as this command is deprecated by Terraform, and they also recommend against using it because it provides no opportunity to review the effects of the operation before updating the state.

upvoted 1 times

✉ **Jayanth** 11 months, 3 weeks ago

B. Configuration file

upvoted 2 times

✉ **Busi57** 11 months, 4 weeks ago

Selected Answer: B

i choose B

upvoted 1 times

✉ **Busi57** 11 months, 4 weeks ago

I choose B

upvoted 1 times

✉ **cdechery** 1 year ago

Selected Answer: B

Of course it is B. Credentials are used during refresh.

upvoted 1 times

 **Ibrahim2023** 1 year ago

The answer is C. Credentials.

When you run terraform refresh, Terraform will read the state file and the configuration file, but it will not read the credentials. This is because credentials are used to connect to the cloud provider, and Terraform does not need to connect to the cloud provider to refresh the state file.

upvoted 3 times

 **HMthyl** 1 year ago

Refresh is about ensuring the state file reflects real world objects: it's not bringing the state file in line with your configuration, it's bringing it into line with the real thing

upvoted 2 times

 **connecttozee** 1 year, 3 months ago

Agree with B. Only configuration file not processed for provisioning infra. Provider & credential need login to infra & state file will overwrite new config current infra

upvoted 2 times

 **SilentMilli** 1 year, 4 months ago

Selected Answer: B

he configuration file is not processed during terraform refresh. The configuration file is only used during the terraform apply or terraform plan commands to determine the desired state of the infrastructure. terraform refresh does not make any changes to the infrastructure, it only updates the state file to reflect the current state of the resources in the cloud provider.

upvoted 3 times

 **Mal_8** 1 year, 4 months ago

Selected Answer: B

The answer is B.

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

Selected Answer: B

The terraform refresh command does not create new resources or modify existing resources. It only updates the state to reflect the current configuration. If you want to create new resources or modify existing resources, you need to use the terraform apply command.

upvoted 1 times

 **Fatouch** 1 year, 8 months ago

there is should be two answers, then B and another one?

upvoted 1 times

 **yogishrb2020** 1 year, 9 months ago

Selected Answer: BD

State file and if credentials stored in the state file gets updated.

I chose B and D

upvoted 1 times

 **Pinky0289** 1 year, 10 months ago

There is no change/impact on the configuration file during the refresh, therefore the answer is B. Configuration file.

upvoted 2 times

Question #16

Topic 1

What information does the public Terraform Module Registry automatically expose about published modules?

- A. Required input variables
- B. Optional inputs variables and default values
- C. Outputs
- D. All of the above
- E. None of the above

Correct Answer: E

Reference:

<https://www.terraform.io/docs/registry/modules/publish.html>

Private Registries

JUMP TO SECTION ▾

The registry at registry.terraform.io only hosts public modules, but most organizations have some modules that can't, shouldn't, or don't need to be public.

You can load private modules [directly from version control and other sources](#), but those sources don't support [version constraints](#) or a browsable marketplace of modules, both of which are important for enabling a producers-and-consumers content model in a large organization.

If your organization is specialized enough that teams frequently use modules created by other teams, you will benefit from a private module registry.

Community vote distribution

D (98%)

2%

✉️  **Egger1992** Highly Voted 2 years, 2 months ago

Selected Answer: D

According to the documentation provided, D is correct.

Extract from documentation:

"The registry extracts information about the module from the module's source. The module name, provider, documentation, inputs/outputs, and dependencies are all parsed and available via the UI or API, as well as the same information for any submodules or examples in the module's source repository."

upvoted 27 times

✉️  **doodlearmy** Most Recent 1 week, 3 days ago

Selected Answer: D

All, According to the documentation provided, D is correct.

upvoted 1 times

✉️  **udenaro** 3 weeks, 6 days ago

Selected Answer: D

the correct answer is D

upvoted 1 times

✉️  **starkonbullet** 1 month, 4 weeks ago

Selected Answer: D

Exposed variables, optional variables & outputs are further given to use them in our custom module.

upvoted 1 times

✉️  **barkadog** 6 months, 1 week ago

Either the person choosing the correct answers on this website was drunk, or this website deliberately wants us to fail.

upvoted 4 times

✉️  **enook** 6 months, 2 weeks ago

Selected Answer: D

DDDDDDDD

upvoted 1 times

-  **[Removed]** 9 months, 3 weeks ago
Definitely D
upvoted 1 times
-  **ktritesh** 10 months, 1 week ago
The question is asking for public, not private, lol D is correct
upvoted 1 times
-  **shivamkr singh** 10 months, 2 weeks ago
I believe the D is the correct answer. The answer marked as correct is for the private modules not for the public modules. It might be a typo or something.
upvoted 1 times
-  **HarshPatel198** 10 months, 3 weeks ago
Selected Answer: D
All of above
upvoted 3 times
-  **Jayanth** 11 months, 3 weeks ago
D. All of the above
upvoted 1 times
-  **Busi57** 11 months, 4 weeks ago
Selected Answer: D
D teams
upvoted 1 times
-  **Shane_C** 1 year ago
Selected Answer: D
D is correct
upvoted 1 times
-  **IK912** 1 year ago
Answer Definitely D
upvoted 1 times
-  **Gaboh97** 1 year, 1 month ago
Selected Answer: D
D. All of the above
upvoted 1 times
-  **gspb** 1 year, 2 months ago
Selected Answer: D
D. All of the above
upvoted 1 times
-  **camps** 1 year, 3 months ago
Selected Answer: D
D. All of the above.

The public Terraform Module Registry automatically exposes information about published modules, including required input variables, optional input variables and their default values, and outputs. This information is displayed on the module's page in the registry, which can be accessed by anyone.

upvoted 4 times

A. True

B. False

Correct Answer: A

Output values are like function return values.

Reference:

<https://www.terraform.io/docs/language/values/locals.html>

<https://www.terraform.io/docs/language/values/outputs.html>

Community vote distribution

A (100%)

✉️  **Bere**  11 months, 3 weeks ago

Selected Answer: A

Code example:

```
variable "input" {
  description = "An input variable"
  default = "Hello"
}
```

```
locals {
  local_value = "${var.input}, World!"
}
```

```
output "greeting" {
  description = "A greeting message"
  value = local.local_value
}
```

Outputs:

```
greeting = "Hello, World!"  
upvoted 8 times
```

✉️  **HarshPatel198**  10 months, 3 weeks ago

Selected Answer: A

Bere is right

upvoted 1 times

✉️  **Jayanth** 11 months, 3 weeks ago

A. True

upvoted 1 times

✉️  **Busi57** 11 months, 4 weeks ago

Selected Answer: A

Teams A

upvoted 1 times

✉️  **VSMu** 1 year ago

Answer A.

```
# Output the local variable as an output value
output "instance_public_ip" {
  value = local.instance_public_ip
}
```

upvoted 1 times

✉  **SammySunny** 1 year ago

Why not B? The documentation says "A local value can only be accessed in expressions within the module where it was declared"
upvoted 1 times

✉  **HMthyl** 1 year ago

Confirmed using a project, A is true. That definition threw me too - but if you explicitly output the local value it can be accessed as expected
upvoted 3 times

✉  **IK912** 1 year ago

Answer Definitely A

upvoted 1 times

✉  **karendavtyan** 1 year, 2 months ago

Selected Answer: A

A. true

upvoted 1 times

✉  **gargaditya** 1 year, 4 months ago

Selected Answer: A

1. Output values make information about your infrastructure available on the command line, and can expose information for other Terraform configurations to use. Output values are similar to return values in programming languages.

2.Basic Syntax:

<https://drive.google.com/file/d/1td5DcyZ-7iHL3Xhpwep4deEEfKbC7DkT/view?usp=sharing>

3.Using Output Values with Modules(logical concept described in diagram below):

<https://drive.google.com/file/d/1tbYWsCSev063ibz9Pv9poqViOY6fAAsG/view?usp=sharing>

Basically,the question is narrowed down to exposing locals from the root module(can be any resource attribute in general).

upvoted 1 times

✉  **SilentMilli** 1 year, 5 months ago

Selected Answer: A

If a module uses local values, those values can be exposed using the "terraform output" directive. By using the terraform output directive, you can define an output for a module that provides information about the local values used within the module. This information can then be consumed by other parts of the Terraform configuration, allowing you to make use of the values that have been set within the module. To expose a local value, you would define an output block in the module, specifying the name and value of the output, and then reference the output in other parts of your Terraform configuration using the "terraform output" data source.

upvoted 3 times

✉  **Tanacet** 1 year, 5 months ago

Selected Answer: A

It is A

upvoted 1 times

✉  **ssanjayt** 1 year, 6 months ago

Selected Answer: A

A is correct

upvoted 1 times

✉  **jj22222** 1 year, 8 months ago

Selected Answer: A

A is correct

upvoted 1 times

✉  **hectordj** 1 year, 9 months ago

Selected Answer: A

A is correct

upvoted 1 times

✉  **poojithakadiyala** 1 year, 9 months ago

A is correct

upvoted 1 times

 **daaww** 1 year, 10 months ago

Selected Answer: A

A is correct

Question #18

Topic 1

You should store secret data in the same version control repository as your Terraform configuration.

A. True

B. False

Correct Answer: B

Reference:

<https://blog.gruntwork.io/a-comprehensive-guide-to-managing-secrets-in-your-terraform-code-1d586955ace1>

Community vote distribution

B (100%)

 **Molly1994** 1 month, 2 weeks ago

You should store secrets in vaults. And retrieve the secrets when used in your terraform application. So the answer is B false.
upvoted 2 times

 **Jayanth** 11 months, 3 weeks ago

B. False
upvoted 1 times

 **tfdestroy** 11 months, 3 weeks ago

Selected Answer: B

B: It is not recommended to store secret data such as passwords, API keys, or other sensitive information in your version control system (VCS). This is a general best practice in software development, not just with Terraform.
upvoted 2 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: B
it's no secure B
upvoted 1 times

 **IK912** 1 year ago

Answer Definitely B
upvoted 1 times

 **karendavtyan** 1 year, 2 months ago

Selected Answer: B
B. False
upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: B
B. False
upvoted 1 times

Storing secret data in the same version control repository as Terraform configuration is not recommended as it increases the risk of exposing sensitive information. Version control systems are designed for sharing and collaboration, which means that they may not have robust security mechanisms to protect against unauthorized access.

upvoted 1 times

 **Nunyabiznes** 1 year, 3 months ago

I know it is not recommended, but technically you really "could" store it. SO, this is kinda confusing since it is not asking for Best Security Practices, but just want to check if it is possible or not
upvoted 1 times

 **Multi_Cloud** 1 year, 3 months ago

If you re-read the question it says, You "should" - which means the answer should be B - false.
Had it been "could/might" your argument would have been correct.
upvoted 2 times

 **SilentMilli** 1 year, 5 months ago

Selected Answer: B

It is generally considered insecure to store secret data, such as passwords, API keys, and other sensitive information, in the same version control repository as your Terraform configuration. This is because version control repositories are often publicly accessible, and if sensitive information is stored in the repository it can be easily accessed by unauthorized individuals. Additionally, version control repositories typically have a history of all changes made to files, so even if sensitive information is deleted at a later point, it can still be retrieved from the repository history. To properly secure secret data, it is recommended to store it in a secure and encrypted format, such as in a secure vault or by using a tool specifically designed for storing secrets.

upvoted 4 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer : false.
upvoted 4 times

 **fabiomlop** 2 years ago

Selected Answer: B

If you store secrets in a version control system, they will be exposed in plain text to anyone with access to the repository.
upvoted 4 times

 **Enrico** 2 years, 2 months ago

Question #19

Topic 1

Which of the following is not a valid string function in Terraform?

- A. split
- B. join
- C. slice
- D. chomp

Correct Answer: D

Reference:

<https://www.terraform.io/docs/language/functions/chomp.html>

`chomp` removes newline characters at the end of a string.

This can be useful if, for example, the string was read from a file that has a newline character at the end.

Examples

```
> chomp("hello\n")
hello
> chomp("hello\r\n")
hello
> chomp("hello\n\n")
hello
```

[Copy !\[\]\(133e5144baf9c90220b67ca9e7f162e0_img.jpg\)](#)

Community vote distribution

C (93%)

7%

✉  **testtaker_1885**  2 years, 2 months ago

answer is c

slice not listed in official docs

<https://www.terraform.io/language/functions>

upvoted 27 times

✉  **bp339** 2 years, 1 month ago

<https://www.terraform.io/language/functions/slice>

upvoted 3 times

✉  **Shane_C** 1 year ago

The question asked for a function that is not a string function.

Slice is a collection function, all other answers are string functions.

Answer is C

upvoted 5 times

✉  **doumx** 2 years, 1 month ago

string function

upvoted 2 times

✉  **samkobadev** 1 year, 5 months ago

so the correct answer is?

upvoted 1 times

✉  **Shane_C** 1 year ago

Answer is C

upvoted 2 times

✉  **coco10k**  2 years, 2 months ago

Selected Answer: C

slice is not in string function list.

upvoted 10 times

✉  **Molly1994**  1 month, 2 weeks ago

answer is C slice. slice is collection function. all other three are string function

upvoted 1 times

✉️ **Venki_dev** 3 months, 1 week ago

Selected Answer: C

slice comes under collective function and not string functions. answer is C

upvoted 1 times

✉️ **samimshaikh** 6 months, 2 weeks ago

Selected Answer: C

slice is collection function and chomp is string function so SLICE is correct answer as not a valid string function "C"

upvoted 1 times

✉️ **MisterROBOT** 8 months, 2 weeks ago

chomp is not a valid string function in Terraform. The other three options are all valid string functions:

split splits a string into a list of strings, based on a delimiter.

join joins a list of strings into a single string, using a separator.

slice extracts a substring from a string, starting at a specified position and ending at a specified position or length.

upvoted 1 times

✉️ **mohamed1999** 6 months, 3 weeks ago

that is not correct :

chomp removes newline characters at the end of a string.

This can be useful if, for example, the string was read from a file that has a newline character at the end.

<https://developer.hashicorp.com/terraform/language/functions/chomp>

upvoted 1 times

✉️ **MisterROBOT** 8 months, 2 weeks ago

ANSWER IS D. Chomp is not valid

Here are some examples of how to use these string functions in Terraform:

```
# Split a string into a list of strings, based on a comma delimiter
variable "tags" {
  type = string
}

output "tags_list" {
  value = split(var.tags, ",")
}

# Join a list of strings into a single string, using a space separator
variable "regions" {
  type = list(string)
}

output "regions_string" {
  value = join(var.regions, " ")
}

# Extract a substring from a string, starting at position 5 and ending at position 10
variable "hostname" {
  type = string
}

output "hostname_suffix" {
  value = slice(var.hostname, 5, 10)
}
```

upvoted 1 times

✉️ **mohamed1999** 6 months, 3 weeks ago

that is not correct :

chomp removes newline characters at the end of a string.

This can be useful if, for example, the string was read from a file that has a newline character at the end.

<https://developer.hashicorp.com/terraform/language/functions/chomp>

upvoted 2 times

✉  **imkhan** 9 months ago

The correct answer is C. Slice function is not listed in the official documentation of terraform string functions
<https://developer.hashicorp.com/terraform/language/functions>
slice is a collection function that extracts some consecutive elements from within a list.
<https://developer.hashicorp.com/terraform/language/functions/slice>

If you are interested to learn about the Terraform string functions you can visit the following link <https://bitsto.cloud/terraform-string-functions>.
upvoted 1 times

✉  **Selvan1978** 9 months, 3 weeks ago

Selected Answer: D

A. split: The split function is used to split a string into a list of substrings based on a specified delimiter.

B. join: The join function is used to concatenate a list of strings into a single string with a specified delimiter.

C. slice: The slice function is used to extract a range of elements from a list or tuple.

D. chomp: The chomp function is not a valid string function in Terraform. It is not part of Terraform's standard function set for working with strings.

So, the correct answer is D. chomp.

upvoted 2 times

✉  **kudakk** 11 months ago

Selected Answer: C

Slice Function is not part of the string function. Others like join, split, chomp are part of it.

upvoted 2 times

✉  **Jayanth** 11 months, 3 weeks ago

C. slice

upvoted 1 times

✉  **Busi57** 11 months, 4 weeks ago

Selected Answer: D

i hesitate between c and D , i choose D

upvoted 1 times

✉  **Venki_dev** 3 months, 1 week ago

slice comes under collective function and not string functions. answer is C

upvoted 1 times

✉  **cdechery** 1 year ago

Slice is not a string function. Why are so many answers here just plain wrong with 99% and sometimes 100% of votes on the correct answer?
upvoted 1 times

✉  **Shane_C** 1 year ago

Selected Answer: C

The question asked for a function that is not a string function.

Slice is a collection function, all other answers are string functions.

Answer is C

upvoted 1 times

✉  **EdwardBHall** 1 year, 1 month ago

Why are the correct answers SO bad in this site? Someone is an absolute idiot.

upvoted 4 times

✉  **babs30** 1 year, 2 months ago

slice is the correct answer, C

upvoted 2 times

✉  **Mehedidevops** 1 year, 2 months ago

why they put in here wrong ans?

upvoted 3 times

Question #20

Topic 1

You have provisioned some virtual machines (VMs) on Google Cloud Platform (GCP) using the gcloud command line tool. However, you are standardizing with

Terraform and want to manage these VMs using Terraform instead.

What are the two things you must do to achieve this? (Choose two.)

- A. Provision new VMs using Terraform with the same VM names
- B. Use the terraform import command for the existing VMs
- C. Write Terraform configuration for the existing VMs
- D. Run the terraform import-gcp command

Correct Answer: *BD*

The terraform import command is used to import existing infrastructure.

Import existing Google Cloud resources into Terraform with Terraformer.

Reference:

<https://www.terraform.io/docs/cli/import/usage.html>

<https://cloud.google.com/docs/terraform>

Community vote distribution

BC (100%)

 **tipzzz** Highly Voted 2 years, 2 months ago
BC for sure

upvoted 32 times

✉  **fabiomlop** Highly Voted 2 years ago

Selected Answer: BC

You should create the equivalent configuration first, and then run import to load it on the state file.

upvoted 17 times

✉  **doodlearmy** Most Recent 1 week, 3 days ago

Selected Answer: BC

there is no 'import-gcp' in terraform official doc

upvoted 1 times

✉  **kingfighters** 3 months, 2 weeks ago

there is no 'import-gcp' in terraform official doc

upvoted 2 times

✉  **enook** 6 months, 2 weeks ago

Selected Answer: BC

BCBCBCBCBCBC

upvoted 2 times

✉  **Tyler2023** 8 months, 4 weeks ago

There are two ways to manage existing resource using Terraform

You can use 'terraform import' or you can use the import block

So answers are:

- B. Use the terraform import command for the existing VMs Most Voted
- C. Write Terraform configuration for the existing VMs

<https://developer.hashicorp.com/terraform/cli/import/usage>

<https://developer.hashicorp.com/terraform/language/import>

upvoted 3 times

✉  **Jayanth** 11 months, 3 weeks ago

- B. Use the terraform import command for the existing VMs
- C. Write Terraform configuration for the existing VMs

upvoted 2 times

✉  **Busi57** 11 months, 4 weeks ago

Selected Answer: BC

i choose BC

upvoted 1 times

✉  **Busi57** 11 months, 4 weeks ago

i choose BC

upvoted 2 times

✉  **Shane_C** 1 year ago

Selected Answer: BC

BC are the correct answers here

upvoted 1 times

✉  **milan92stankovic** 1 year, 1 month ago

Selected Answer: BC

Write configuration and import.

upvoted 1 times

✉  **Ni33** 1 year, 2 months ago

Selected Answer: BC

B&C are correct

upvoted 2 times

 **karendavtyan** 1 year, 2 months ago

Selected Answer: BC

B.True

C. True

upvoted 1 times

 **AzRNoob** 1 year, 3 months ago

BC are the correct options:

B. Use the terraform import command for the existing VMs. This command allows you to import existing infrastructure into your Terraform state file so that Terraform can manage it. You will need to provide the resource type and name, along with any required attributes, for each VM you want to import.

C. Write Terraform configuration for the existing VMs. Once the VMs have been imported into the Terraform state file, you will need to write configuration code that describes the desired state of the VMs. This will typically involve creating a new Terraform module or modifying an existing one to include the imported resources.

Option A is incorrect because provisioning new VMs with the same names would create duplicate resources and could cause conflicts with the existing VMs.

Option D is also incorrect because there is no terraform import-gcp command in Terraform. The correct command for importing GCP resources is simply terraform import.

upvoted 10 times

 **connecttozee** 1 year, 3 months ago

BC is correct

<https://developer.hashicorp.com/terraform/tutorials/state/state-import>

upvoted 1 times

 **Power123** 1 year, 3 months ago

B,C - first write the configuration and then import

upvoted 1 times

 **acheiron** 1 year, 4 months ago

Selected Answer: BC

Write configuration then import the existing infrastructure

Question #21

Topic 1

You have recently started a new job at a retailer as an engineer. As part of this new role, you have been tasked with evaluating multiple outages that occurred during peak shopping time during the holiday season. Your investigation found that the team is manually deploying new compute instances and configuring each compute instance manually. This has led to inconsistent configuration between each compute instance.

How would you solve this using infrastructure as code?

- A. Implement a ticketing workflow that makes engineers submit a ticket before manually provisioning and configuring a resource
- B. Implement a checklist that engineers can follow when configuring compute instances
- C. Replace the compute instance type with a larger version to reduce the number of required deployments
- D. Implement a provisioning pipeline that deploys infrastructure configurations committed to your version control system following code reviews

Correct Answer: A

Community vote distribution

D (99%)

 **tipzzz** Highly Voted 2 years, 2 months ago

Selected Answer: D

D for sure

upvoted 26 times

✉️  **Eniras** 2 years, 2 months ago

Yes Answer is D

upvoted 3 times

✉️  **fabiomlop**  2 years ago

Selected Answer: D

Obviously D... who picks those answers, FFS...

upvoted 16 times

✉️  **SilentH**  3 months ago

Selected Answer: D

C'mon, the answer is obviously D. I'm only on question 21 and have seen at least 5 obviously wrong answers. It's like ExamTopics is trying to people.

upvoted 4 times

✉️  **qs5dq64s5dq5** 8 months, 3 weeks ago

Selected Answer: D

D for sure

upvoted 1 times

✉️  **shivamkrsingh** 10 months, 2 weeks ago

I think the answer is D. The person who stored the A as correct answer might not want to write terraform code or CICD.😊

upvoted 6 times

✉️  **pradeepukg** 11 months ago

Selected Answer: D

D is correct answer

upvoted 1 times

✉️  **jitusonawane89** 11 months, 3 weeks ago

D is correct

upvoted 1 times

✉️  **wheelan** 12 months ago

Selected Answer: D

D - in line with IaC

upvoted 1 times

✉️  **Shane_C** 1 year ago

Selected Answer: D

D is the only thing close to a correct answer here

upvoted 1 times

✉️  **DuboisNicolasDuclair** 1 year ago

Selected Answer: A

Because the topic said *a sample main.tf* it means that we can have many types of files like *.tf

upvoted 1 times

✉️  **HMytl** 12 months ago

This implies the question has completely changed over the last 3 weeks, huh?

upvoted 2 times

✉️  **IK912** 1 year ago

Definitely D

upvoted 1 times

✉️  **Ni33** 1 year, 2 months ago

Selected Answer: D

D is the correct answer

upvoted 1 times

 **karendavtyan** 1 year, 2 months ago

Selected Answer: D

D. for sure

upvoted 1 times

 **rookie1025** 1 year, 2 months ago

man... obviously the answer is D.

So many incorrect answers are there in all the dumps. With this website being so inconsistent with their answers they sure are shameless giving us captcha challenge and asking us to buy the pro version every 5 questions.

upvoted 8 times

 **connecttozee** 1 year, 3 months ago

agree D

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: D

D. Implement a provisioning pipeline that deploys infrastructure configurations committed to your version control system following code review

To solve this problem, you should consider implementing infrastructure as code (IaC) using a provisioning pipeline. By doing so, you can define your compute instances and other infrastructure resources using code, which can be version controlled, reviewed, and tested before being deployed to production. This would eliminate the need for manual configuration and reduce the risk of inconsistent configuration between compute instances.

upvoted 3 times

 **Power123** 1 year, 3 months ago

Correct ans is D

upvoted 1 times

Question #22

Topic 1

terraform init initializes a sample main.tf file in the current directory.

A. True

B. False

Correct Answer: A

Community vote distribution

B (72%)

A (28%)

 **Jlee7**  1 year ago

The answer is False. The terraform init command does not initialize a sample main.tf file in the current directory. It initializes the working directory by downloading and installing provider plugins, downloading modules, and creating a Terraform state file.

If you do not have a main.tf file in your working directory, Terraform will create one for you. However, the file will be empty. You will need to add your Terraform configuration to the main.tf file.

You can find more information about the terraform init command in the Terraform documentation:

<https://www.terraform.io/docs/cli/commands/init.html>

upvoted 16 times

👤 vibzr2023 3 months, 2 weeks ago

The answer is False but terraform init will not create a empty main.tf file but will still initialize the directory for use with Terraform, setting up necessary internal data structures and downloading providers specified in any Terraform configuration files that are present. If you run terraform init in an empty directory or a directory without any .tf files, Terraform will complete the initialization process but will indicate that configuration files were found. You'll need to create main.tf or other .tf files manually. I ran a test

PS C:\Users\rajna\OneDrive\Documents\TerraformCI> terraform init
Terraform initialized in an empty directory!

The directory has no Terraform configuration files. You may begin working with Terraform immediately by creating Terraform configuration files.

PS C:\Users\rajna\OneDrive\Documents\TerraformCI>

upvoted 2 times

👤 aanataliya Highly Voted 10 months, 2 weeks ago

Selected Answer: A

Answer is A. See below points.

1. Question does not ask if it creates main.tf or not. it asks wheather main.tf initializes or not.
2. Ran terraform init in empty directory and it warns "The directory has no Terraform configuration files." it means it tries to find configuration to initialize.
3. configuration file can be of any name with .tf extension.
4. ran command with main.tf in directory with extra curly braces and initialization fails. because it try to initialize main.tf
5. rename main.tf with sample.tf with extra curly braces and it still gives same error.
6. Official documentation says "command initializes a working directory containing Terraform configuration files". main.tf is a configuration file Again, question never says it creates it. question clearly asks if main.tf is initialized. it means main.tf is already exists w.r.t. this question Reference: <https://developer.hashicorp.com/terraform/cli/commands/init>

Please vote for this correct answer

upvoted 7 times

👤 aanataliya 10 months, 2 weeks ago

Those who are thinking its about creating main.tf. Question should have been "terrafrom init creates sample init file" not initializes.

upvoted 2 times

👤 Tyler2023 8 months, 4 weeks ago

The answer is B

The question says terraform init initializes a sample main.tf file,

think about the word "sample" meaning it creates the file which is terraform will not create that for you

By default, terraform init assumes that the working directory already contains a configuration and will attempt to initialize that configuration will never try to create a sample file with sample configuration for you. But the question says it creates that sample file, meaning the directory is empty.

upvoted 1 times

👤 Molly1994 Most Recent 1 month, 1 week ago

main.tf is an option file. it might not exist. terraform init means to download all basic infrastructures from the configuration file. B false.

upvoted 1 times

👤 SilentH 3 months ago

Selected Answer: B

Even if the answer is somehow A, I'm voting B on principle.

upvoted 2 times

👤 DianaPopal 6 months, 2 weeks ago

The correct answer is B. False.

The terraform init command in Terraform is used to initialize a Terraform configuration in the current directory. It downloads the necessary provider plugins and sets up the backend configuration.

However, terraform init does not create a sample main.tf file in the current directory. The main.tf file is the main configuration file in Terraform where you define your infrastructure resources and their configurations. It is typically created manually by the user and contains the desired state of the infrastructure.

When you run terraform init, it looks for an existing main.tf file in the current directory to load the configuration. If the main.tf file does not exist, Terraform will not create it automatically.

upvoted 1 times

👤 Hp45 10 months, 2 weeks ago

A lot of people are taking this question as init command is creating a main.tf. The context in the question is - if there was a sample main.tf in current folder then init would initialize the file. I vote for A.

upvoted 4 times

✉  **Shane_C** 1 year ago

It absolutely does not generate a main.tf; B is the correct answer

upvoted 2 times

✉  **DuboisNicolasDuclair** 1 year ago

A

Because terraform initialize *a sample main.tf* it means that we can have a multiple types of files like *.tf

So A is a good answer

upvoted 2 times

✉  **Shane_C** 1 year ago

It absolutely does not generate a main.tf; B is the correct answer

upvoted 2 times

✉  **karendavtyan** 1 year, 2 months ago

Selected Answer: B

B. False

upvoted 1 times

✉  **Ni33** 1 year, 2 months ago

Selected Answer: B

B is correct. It initializes versions and providers irrespective of name of the tf files.

upvoted 2 times

✉  **Rezi** 1 year, 3 months ago

A is correct 100%

terraform init initializes any file with the extension ".tf" in the pwd. it says "a" sample main.tf, it didn't say "the" sample main.tf. Please note the difference.

upvoted 2 times

✉  **AzRNoob** 1 year, 3 months ago

B. False.

The terraform init command initializes a new or existing Terraform working directory by downloading and installing any necessary plugins and modules specified in the configuration. It does not create a main.tf file or any other configuration file in the current directory.

You will need to create a main.tf file (or another configuration file with a different name) manually and write the necessary configuration code for your infrastructure. Once you have done so, you can run terraform init to prepare the working directory for use with Terraform.

upvoted 2 times

✉  **Faaizz** 1 year, 3 months ago

Selected Answer: B

BBBBBBBB

upvoted 1 times

✉  **Power123** 1 year, 3 months ago

B for sure

upvoted 1 times

✉  **sagunala5** 1 year, 3 months ago

By default, terraform init assumes that the working directory already contains a configuration and will attempt to initialize that configuration.

upvoted 2 times

 **Nunyabiznes** 1 year, 3 months ago

Selected Answer: B

The terraform init command initializes a new or existing Terraform working directory by downloading and installing any required providers and modules, as well as setting up the backend. It does not create a sample main.tf file in the current directory. The main.tf file is a configuration file where you define the resources that you want to create using Terraform. It needs to be created manually by the user.

upvoted 2 times

 **sagunala5** 1 year, 3 months ago

who asked whether sample main.tf was created, they are asking if it is initialising.
so my answer is A

upvoted 1 times

Question #23

Topic 1

Which two steps are required to provision new infrastructure in the Terraform workflow? (Choose two.)

- A. Destroy
- B. Apply
- C. Import
- D. Init
- E. Validate

Correct Answer: BD

Reference:

<https://www.terraform.io/guides/core-workflow.html>

Community vote distribution

BD (100%)

 **enook** 6 months, 2 weeks ago

Selected Answer: BD

BD for sure

upvoted 2 times

 **hungryan91** 1 year ago

Selected Answer: BD

Definitely BD

upvoted 1 times

 **Jlee7** 1 year ago

Glad to see most popular answer and examtopics answer agree!! :) B&D

upvoted 3 times

 **karendavtyan** 1 year, 2 months ago

Selected Answer: BD

B and D

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: BD

B and D

upvoted 1 times

 **connecttozee** 1 year, 3 months ago

required sure init first & apply later

BD is correct

upvoted 2 times

✉️  **Power123** 1 year, 3 months ago

B & D - init and apply

upvoted 1 times

✉️  **Nunyabiznes** 1 year, 3 months ago

Selected Answer: BD

The two steps required to provision new infrastructure in the Terraform workflow are:

B. Apply: This step applies the changes to the infrastructure by creating or modifying resources in accordance with the configuration described in the Terraform code.

D. Init: This step initializes a new or existing Terraform working directory by downloading and installing any required providers and modules, as well as setting up the backend.

The other options are:

A. Destroy: This step destroys the resources that are managed by Terraform in the specified configuration.

C. Import: This step allows the user to import existing infrastructure into Terraform's state.

E. Validate: This step checks the syntax and validity of the Terraform configuration files without actually creating or modifying any resources.

upvoted 3 times

✉️  **BilalIg93350** 1 year, 4 months ago

The two required steps to provision new infrastructure in the Terraform workflow are:

B. Apply - This step applies the changes to the infrastructure by creating, modifying, or deleting resources based on the Terraform configuration.

D. Init - This step initializes a new or existing Terraform configuration by downloading the required provider plugins and setting up the backend to store the state of the infrastructure.

Options A, C, and E are not required steps to provision new infrastructure in the Terraform workflow:

Destroy (option A) is used to destroy the resources created by the Terraform configuration, which is not required to provision new infrastructure. Import (option C) is used to import an existing infrastructure resource into the Terraform state, which is not a required step to provision new infrastructure.

Validate (option E) is used to validate the syntax and structure of the Terraform configuration but is not a required step to provision new infrastructure.

upvoted 2 times

✉️  **SilentMilli** 1 year, 5 months ago

Selected Answer: BD

The Terraform workflow consists of several steps for provisioning new infrastructure. The two steps that are required to provision new infrastructure in the Terraform workflow are:

B. Apply: The "terraform apply" command is used to provision new infrastructure. This command takes the Terraform configuration files as input and creates or updates the resources specified in the configuration files.

D. Init: The "terraform init" command is used to initialize a Terraform configuration directory. This command must be run before running the "terraform apply" command to ensure that the required Terraform modules and plugins are installed and available.

Note that other steps in the Terraform workflow, such as "terraform destroy" or "terraform import," may also be used depending on your specific needs and the nature of your infrastructure. However, the "terraform apply" and "terraform init" commands are the two steps that are required to provision new infrastructure in the Terraform workflow.

upvoted 1 times

✉️  **awsguys** 1 year, 5 months ago

B , D right

upvoted 1 times

✉️  **FarziWaliMarzi** 1 year, 7 months ago

Selected Answer: BD

Obviously, plan can be skipped.

upvoted 1 times

 **RVivek** 1 year, 9 months ago

Selected Answer: BD

A- is to destroy
C- is to import an existing resource that was manually created . The question is about provisioning a new resource
E - is just to validate the command structure

upvoted 1 times

 **cjig** 1 year, 11 months ago

Selected Answer: BD

Options B and D are correct.

upvoted 3 times

 **Eltooth** 2 years ago

Selected Answer: BD

B and D are correct answers.
upvoted 3 times

 **BlackZeros** 2 years ago

Selected Answer: BD

b and d sounds logical
upvoted 2 times

 **javidabillo** 2 years, 1 month ago

Selected Answer: BD

b and d
upvoted 3 times

Question #24

Topic 1

Why would you use the terraform taint command?

- A. When you want to force Terraform to destroy a resource on the next apply
- B. When you want to force Terraform to destroy and recreate a resource on the next apply
- C. When you want Terraform to ignore a resource on the next apply
- D. When you want Terraform to destroy all the infrastructure in your workspace

Correct Answer: B

The terraform taint command manually marks a Terraform-managed resource as tainted, forcing it to be destroyed and recreated on the next apply.

Reference:

<https://www.terraform.io/docs/cli/commands/taint.html>

Community vote distribution

B (100%)

 **Eltooth**  2 years ago

Selected Answer: B

Noe of the answers really explain what taint command does - except B is closest.

"The terraform taint command informs Terraform that a particular object has become degraded or damaged. Terraform represents this by marking the object as "tainted" in the Terraform state, and Terraform will propose to replace it in the next plan you create."

<https://www.terraform.io/cli/commands/taint>

upvoted 10 times

✉️  **Alandt** Most Recent 5 months, 3 weeks ago

Selected Answer: B

IMPORTANT NOTE: taint command is deprecated. For Terraform v0.15.2 and later, Hashicorp recommends using the -replace option with terraform apply instead

upvoted 4 times

✉️  **Jerry_m10** 7 months, 3 weeks ago

Outdated question for v3.0.0 test.

Although B is correct, but we now have terraform -replace="resource_name" for this now.

upvoted 2 times

✉️  **Halimb** 10 months, 2 weeks ago

Selected Answer: B

Although B is correct here, the question is old and the command is deprecated. Instead use terraform apply -replace="". See <https://developer.hashicorp.com/terraform/cli/commands/taint>

upvoted 2 times

✉️  **karendavtyan** 1 year, 2 months ago

Selected Answer: B

B. When you want to force Terraform to destroy and recreate a resource on the next apply

upvoted 1 times

✉️  **Ni33** 1 year, 2 months ago

Selected Answer: B

B is correct

upvoted 1 times

✉️  **Power123** 1 year, 3 months ago

This command is now deprecated and suggested option is now terraform apply -replace="state_object" . Ans B

upvoted 4 times

✉️  **Bilalg93350** 1 year, 4 months ago

The correct answer is C. When you want Terraform to ignore a resource on the next apply.

The terraform taint command is used to mark a resource managed by Terraform as "tainted," which means that it needs to be recreated on the next apply. This is useful when a resource is in an inconsistent state or needs to be recreated for some other reason.

However, the taint command does not destroy or force Terraform to destroy a resource on the next apply (option A and B). Instead, it marks the resource as tainted, indicating that it needs to be recreated. When you run terraform apply after marking a resource as tainted, Terraform will destroy and recreate the resource.

Option D is incorrect because the terraform taint command only marks a single resource as tainted and does not destroy all the infrastructure in your workspace.

upvoted 1 times

✉️  **SilentMilli** 1 year, 5 months ago

Selected Answer: B

The "terraform taint" command is used to mark a resource as "tainted," indicating to Terraform that the resource should be destroyed and recreated on the next "terraform apply" command. This can be useful in cases where you need to make changes to a resource that cannot be updated in place, or if you want to enforce a clean rebuild of a resource for some other reason.

It's important to note that tainting a resource will not immediately destroy the resource, but will instead cause Terraform to plan to destroy and recreate the resource on the next "terraform apply" command. This allows you to review and confirm the changes that Terraform will make before actually applying them.

upvoted 2 times

✉️  **BlackZeros** 2 years ago

Selected Answer: B

b it is

upvoted 3 times

 **javibadillo** 2 years, 1 month ago

Selected Answer: B

<https://www.terraform.io/cli/commands/taint#recommended-alternative>

upvoted 3 times

 **softarts** 2 years, 2 months ago

it is B

upvoted 4 times

 **d0ug7979** 2 years, 2 months ago

This command is now deprecated since v0.15.2 and suggested option is now terraform apply -replace="state_object"
<https://www.terraform.io/cli/commands/taint#recommended-alternative>

upvoted 3 times

 **biscuithammer** 2 years, 1 month ago

yes but on the exam it's still a valid answer

upvoted 1 times

Question #25

Topic 1

Terraform requires the Go runtime as a prerequisite for installation.

A. True

B. False

Correct Answer: B

Reference:

<https://www.terraform.io/docs/extend/guides/v1-upgrade-guide.html>

As of September 2019, Terraform provider developers importing the Go module

`github.com/hashicorp/terraform`, known as Terraform Core, should switch to

`github.com/hashicorp/terraform-plugin-sdk`, the Terraform Plugin SDK, instead.

Why a separate module?

While the `helper/*` and other packages in Terraform Core has served us well, in order for provider development to evolve, the SDK needed to break out into its own repository.

Terraform Core's versioning has been oriented towards practitioners. With the "unofficial" SDK existing in the core repository, the SDK becomes tied to Core releases and cannot follow semantic versioning. The new standalone SDK github.com/hashicorp/terraform-plugin-sdk follows semantic versioning starting with v1.0.0.

We will use the term "legacy Terraform plugin SDK" when referring to the version of Terraform Core imported and used by providers.

Community vote distribution

 **Empel** Highly Voted 2 years ago

Selected Answer: B

Terraform software is written in Go but it is not required to have Go installed as you get a binary after installing terraform.
upvoted 11 times

 **wsyh** Most Recent 1 year, 2 months ago

Selected Answer: B

Vote B

upvoted 2 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

BBBBBB

upvoted 2 times

 **karendavtyan** 1 year, 2 months ago

Selected Answer: B

B. False

upvoted 2 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

B is correct

upvoted 1 times

 **Power123** 1 year, 3 months ago

Ans is B. Terraform is written in Go programming language and doesn't require Go runtime

upvoted 1 times

 **SilentMilli** 1 year, 5 months ago

Selected Answer: B

Terraform is written in Go programming language, but it does not require the Go runtime to be installed on the machine where Terraform is installed. You only need to have the Terraform binary installed to be able to use Terraform to provision infrastructure. You can download the appropriate binary for your operating system from the Terraform website, and then add it to your PATH so that you can run Terraform from the command line.

upvoted 3 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer : False.

This has been updated with <https://www.terraform.io/plugin/sdkv2/guides/v1-upgrade-guide> and <https://www.terraform.io/plugin/sdkv2/guides/v2-upgrade-guide>

upvoted 4 times

 **amrith501** 2 years, 1 month ago

Selected Answer: B

I did not see any documentation saying terraform requires Go runtime. I will go with false

upvoted 3 times

- A. You see a status message that you cannot acquire the lock
- B. You have a high priority change
- C. Automatic unlocking failed
- D. You apply failed due to a state lock

Correct Answer: C

Manually unlock the state for the defined configuration.

Reference:

<https://www.terraform.io/docs/cli/commands/force-unlock.html>

Community vote distribution

C (100%)

 **Egger1992**  2 years, 2 months ago

Selected Answer: C

Be very careful with this command. If you unlock the state when someone else is holding the lock it could cause multiple writers. Force unlock should only be used to unlock your own lock in the situation where automatic unlocking failed.

Source: <https://www.terraform.io/language/state/locking>

upvoted 23 times

 **SilentMilli**  1 year, 4 months ago

Question #27

Topic 1

an
ng

Terraform can import modules from a number of sources `` which of the following is not a valid source?

- A. FTP server
- B. GitHub repository
- C. Local path
- D. Terraform Module Registry

Correct Answer: A

Community vote distribution

A (100%)

 **karendavtyan** 1 year, 2 months ago

Selected Answer: A

A. FTP server

upvoted 3 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

A is the correct answer.

upvoted 1 times

 **Bluemoon22** 1 year, 2 months ago

A, ftp server

upvoted 1 times

 **AzRNoob** 1 year, 3 months ago

A. FTP server

Question #28

Topic 1

Which of the following is available only in Terraform Enterprise or Cloud workspaces and not in Terraform CLI?

- A. Secure variable storage
- B. Support for multiple cloud providers
- C. Dry runs with terraform plan
- D. Using the workspace as a data source

Correct Answer: A

Community vote distribution

A (91%)

9%

 **koneba1309**  2 years, 2 months ago

Selected Answer: A

I think it is A.

upvoted 10 times

 **DianaPopal**  6 months, 2 weeks ago

The correct answer is A. Secure variable storage.

Secure variable storage is a feature that is available only in Terraform Enterprise or Terraform Cloud workspaces, and it is not available in the Terraform CLI.

Secure variable storage allows you to store sensitive information, such as API keys, passwords, or other credentials, securely within Terraform. These variables are encrypted and protected, ensuring that they are not exposed in plain text.

upvoted 7 times

 **IK912**  1 year ago

Definitely A

upvoted 1 times

 **Ni33** 1 year, 2 months ago

A is correct answer. Terraform Cloud do provide secure storage for variables.

upvoted 3 times

 **Bluemoon22** 1 year, 2 months ago

Answer is A, secure variable storage

upvoted 1 times

 **AzRNoob** 1 year, 3 months ago

A. Secure variable storage is available only in Terraform Enterprise or Cloud workspaces and not in Terraform CLI.

Terraform Enterprise and Cloud workspaces provide additional features and functionality beyond what is available in the open-source Terraform CLI. One of these features is the ability to store sensitive data, such as API keys or passwords, securely within a workspace. This allows team members to share infrastructure code without exposing sensitive information.

upvoted 3 times

 **Power123** 1 year, 3 months ago

Correct answer is A

upvoted 1 times

 **r1ck** 1 year, 4 months ago

<https://developer.hashicorp.com/terraform/tutorials/configuration-language/sensitive-variables>

still A ?

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

Selected Answer: A

Its A indeed

upvoted 1 times

 **RVivek** 1 year, 9 months ago

Selected Answer: A

A is available in Terraform cloud and not in Terraform OSS

upvoted 1 times

 **Optimus** 1 year, 10 months ago

Selected Answer: A

It is definitely A

upvoted 4 times

 **yuvifose** 1 year, 12 months ago

Selected Answer: A

I think is A

upvoted 3 times

 **nhatne** 2 years ago

Selected Answer: A

it's A

upvoted 4 times

 **Ahmad_Terraform** 2 years ago

i think A

upvoted 1 times

 **bp339** 2 years, 1 month ago

Selected Answer: A

A is correct

upvoted 3 times

 **biscuithammer** 2 years, 1 month ago

Selected Answer: A

it should be A

upvoted 2 times

 **ItaloVinodi** 2 years, 2 months ago

Selected Answer: A

I think A is correct

upvoted 2 times

terraform validate validates the syntax of Terraform files.

- A. True
- B. False

Correct Answer: A

The terraform validate command validates the syntax and arguments of the Terraform configuration files.

Reference:

<https://www.terraform.io/docs/cli/code/index.html>

Community vote distribution

A (100%)

 **kilowd** Highly Voted 1 year, 10 months ago

Validate runs checks that verify whether a configuration is syntactically valid and internally consistent, regardless of any provided variables or existing state. It is thus primarily useful for general verification of reusable modules, including correctness of attribute names and value types.
upvoted 8 times

 **enook** Most Recent 6 months, 2 weeks ago

Selected Answer: A

AAAAAAA

upvoted 1 times

 **Caput** 11 months ago

Selected Answer: A

It's true.

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

A is the correct answer

upvoted 1 times

 **Bluemoon22** 1 year, 2 months ago

answer is A, true

upvoted 1 times

 **Power123** 1 year, 3 months ago

Ans is A it validates the syntax

upvoted 1 times

 **Eltooth** 2 years ago

Selected Answer: A

A is correct answer.

<https://www.terraform.io/cli/commands/validate>

upvoted 2 times

 **biscuithammer** 2 years, 1 month ago

Selected Answer: A

A is correct

<https://www.terraform.io/cli/commands/validate>

upvoted 4 times

Question #30

Topic 1

You have used Terraform to create an ephemeral development environment in the cloud and are now ready to destroy all the infrastructure described by your

Terraform configuration. To be safe, you would like to first see all the infrastructure that will be deleted by Terraform.

Which command should you use to show all of the resources that will be deleted? (Choose two.)

- A. Run `terraform plan -destroy`.
- B. This is not possible. You can only show resources that will be created.
- C. Run `terraform state rm *`.
- D. Run `terraform destroy` and it will first output all the resources that will be deleted before prompting for approval.

Correct Answer: CD

Reference:

<https://www.terraform.io/docs/cli/commands/state/rm.html>

Command: state rm

JUMP TO SECTION ▾

The main function of [Terraform state](#) is to track the bindings between resource instance addresses in your configuration and the remote objects they represent. Normally Terraform automatically updates the state in response to actions taken when applying a plan, such as removing a binding for a remote object that has now been deleted.

You can use `terraform state rm` in the less common situation where you wish to remove a binding to an existing remote object without first destroying it, which will effectively make Terraform "forget" the object while it continues to exist in the remote system.

Community vote distribution

AD (94%)

6%

✉️  **Ipergorta** Highly Voted 2 years, 2 months ago

The answer is A,D
upvoted 23 times

✉️  **ItaloVinodi** Highly Voted 2 years, 2 months ago

Selected Answer: AD
We need to choose two so AD
upvoted 12 times

✉️  **ravi135** Most Recent 5 months ago

answer AD
upvoted 1 times

✉️  **Shane_C** 1 year ago

Selected Answer: AD
AD are correct
upvoted 2 times

✉️  **vj_dhaksh** 1 year, 1 month ago

Answer is AD
C - incorrect as running "terraform state rm **" throws error stating variable required (* wont pick all)
upvoted 3 times

✉️  **karendavtyan** 1 year, 2 months ago

D. Run terraform destroy and it will first output all the resources that will be deleted before prompting for approval.
upvoted 1 times

✉️  **karendavtyan** 1 year, 2 months ago

Selected Answer: A
A. Run terraform plan -destroy.
upvoted 1 times

✉️  **Ni33** 1 year, 2 months ago

Selected Answer: AD
A and D are correct.
upvoted 1 times

✉️  **Bluemoon22** 1 year, 2 months ago

A is only to show the resources will be destroyed, terraform plan -destroy
D is a common operation from terraform destroy
upvoted 1 times

 **AzRNoob** 1 year, 3 months ago

A. Run terraform plan -destroy.

This command will generate a plan that shows all of the changes Terraform will make to the infrastructure, including any resources that will be destroyed. The -destroy flag specifies that only the changes that will result in the destruction of resources should be shown.

Option D is also partially correct, as running terraform destroy will output a list of resources that will be destroyed before prompting for approval. However, it is generally recommended to run terraform plan -destroy first to preview the changes before actually executing them with terraform destroy.

Option B is incorrect, as it is possible to show resources that will be deleted using the terraform plan -destroy command.

Option C is also incorrect, as running terraform state rm * will remove all resources from the Terraform state file, effectively "forgetting" about them and making it impossible to manage them with Terraform.

upvoted 6 times

 **Faaizz** 1 year, 3 months ago

Selected Answer: AD

A & D are correct here

upvoted 1 times

 **connecttozee** 1 year, 3 months ago

100% is AD

upvoted 1 times

 **Power123** 1 year, 3 months ago

A and D are correct

upvoted 1 times

 **r1ck** 1 year, 4 months ago

<https://developer.hashicorp.com/terraform/cli/commands/destroy>

AD

upvoted 1 times

 **andersonbispos42** 1 year, 6 months ago

Selected Answer: AD

The answer is A, D

upvoted 1 times

 **andersonbispos42** 1 year, 6 months ago

Selected Answer: A

The answer is only A

upvoted 1 times

 **vikramv1r** 1 year, 7 months ago

Correct answer - AD

upvoted 1 times

Question #31

Topic 1

Which of the following is the correct way to pass the value in the variable num_servers into a module with the input servers?

- A. servers = num_servers
- B. servers = variable.num_servers
- C. servers = var(num_servers)
- D. servers = var.num_servers

Correct Answer: A

Community vote distribution

D (100%)

tipzzz [Highly Voted] 2 years, 2 months ago

D for sure

upvoted 22 times

vitasac [Highly Voted] 2 years, 2 months ago

Selected Answer: D

Yes I'm agree D

upvoted 10 times

Cololand [Most Recent] 2 months, 1 week ago

D - Who even marks the right answers at examtopics?

upvoted 1 times

enook 6 months, 2 weeks ago

Selected Answer: D

DDDDDDD

upvoted 1 times

ghostGuiggs 8 months, 2 weeks ago

Selected Answer: D

D is the answer

upvoted 1 times

Kizerfor 10 months, 3 weeks ago

D for sure

upvoted 1 times

Bere 11 months, 3 weeks ago

Selected Answer: D

In Terraform, variables are referred to using the syntax var.<variable_name>. Here, var is a special namespace that contains all input variables

For example, if you have a variable named num_servers in your configuration and you want to pass it to a module, you would do it like this:

```
variable "num_servers" {
  description = "The number of servers to create"
  default = 5
}
```

```
module "server_module" {
  source = "app.terraform.io/example/server"
  version = "1.0.0"

  servers = var.num_servers
}
```

In the above example, the input variable num_servers is defined with a default value of 5. Then, the server_module is being called and the servers argument is being set to the value of var.num_servers, which is 5.

upvoted 7 times

Shane_C 1 year ago

Selected Answer: D

D is correct

upvoted 1 times

Ni33 1 year, 2 months ago

Selected Answer: D

D is the one

upvoted 1 times

✉  **Bluemoon22** 1 year, 2 months ago
D, servers = var.num_servers
upvoted 2 times

✉  **Power123** 1 year, 3 months ago
Ans is D
upvoted 1 times

✉  **thor7** 1 year, 3 months ago
Selected Answer: D
var.num_servers is correct and full answer
upvoted 2 times

✉  **Power123** 1 year, 3 months ago
D is the ans
upvoted 1 times

✉  **kartikjena31** 1 year, 6 months ago
I also go with D
upvoted 2 times

✉  **lalgebalal** 1 year, 8 months ago
D for Sure
upvoted 1 times

✉  **RVivek** 1 year, 9 months ago

Selected Answer: D
D
to pass a value to a module the syntax is
<child module variable name > = <value>
Here child module variable name is servers
to reference the value of a variable in main/root module var. <name_of _the varibale>
upvoted 3 times

✉  **najsljedi** 1 year, 10 months ago

Selected Answer: D
D correct
upvoted 2 times

Question #32

Topic 1

A Terraform provisioner must be nested inside a resource configuration block.

- A. True
- B. False

Correct Answer: A

Most provisioners require access to the remote resource via SSH or WinRM, and expect a nested connection block with details about how to connect.

Reference:

<https://www.terraform.io/docs/language/resources/provisioners/connection.html>

Community vote distribution

A (65%)

B (35%)

✉  **quixo** Highly Voted 2 years ago
A: True.

Example:

```
resource "aws_instance" "web" {
# ...

provisioner "local-exec" {
command = "echo The server's IP address is ${self.private_ip}"
}
}

upvoted 14 times
```

✉️  **kiran15789** Highly Voted 1 year, 2 months ago

Selected Answer: A

To use a provisioner in Terraform, you must include it as part of a resource configuration block. This is because a provisioner operates on a resource that has been created or updated by Terraform.

upvoted 5 times

✉️  **090200f** Most Recent 5 days, 16 hours ago

Answer is A: True

Multiple provisioners can be specified within a resource. Multiple provisioners are executed in the order they're defined in the configuration file
<https://developer.hashicorp.com/terraform/language/resources/provisioners/syntax>

```
resource "aws_instance" "web" {
# ...
```

```
provisioner "local-exec" {
command = "echo first"
}
```

```
provisioner "local-exec" {
command = "echo second"
}
}
```

upvoted 1 times

✉️  **090200f** 5 days, 16 hours ago

<https://developer.hashicorp.com/terraform/language/resources/provisioners/connection>

Connection blocks don't take a block label and can be nested within either a resource or a provisioner.

A connection block nested directly within a resource affects all of that resource's provisioners.

A connection block nested in a provisioner block only affects that provisioner and overrides any resource-level connection settings.
in the question they asked 'must' so may be the answer is False : B . I think so please confirm. I am confused now.

upvoted 1 times

✉️  **ravk2321** 4 months, 1 week ago

Selected Answer: A

No Standalone Provisioners: Terraform does not support defining provisioners as standalone elements outside of resource blocks. All provisioner configurations must be part of a resource definition because their execution context is tightly coupled with resources.

Alternatives to Provisioners: For scenarios where you might think about using provisioners in a more decoupled or independent manner, Terraform suggests other mechanisms. For instance, if the goal is to manage configuration or orchestration that seems beyond the scope of provisioners, it should be tightly coupled to a single resource's lifecycle, tools like Ansible, Chef, Puppet, or Terraform itself (through resource dependencies or proper module design) are recommended to handle such configurations. These tools can be used in conjunction with Terraform but managed through their mechanisms for orchestration or configuration management, rather than through Terraform provisioners.

upvoted 1 times

✉️  **samimshaikh** 6 months, 2 weeks ago

Selected Answer: A

True provisioner should only placed under the resource block

PS E:\Terraform> terraform.exe validate

```
| Error: Unsupported block type
| on main.tf line 34:
| 34: provisioner "local-exec" {
|
| Blocks of type "provisioner" are not expected here.
```

upvoted 1 times

✉️  **gold4otas** 6 months, 2 weeks ago

Selected Answer: B

While it's common to place provisioners inside the resource block for clarity and organization, they can also be defined outside of the block at same level. Both approaches are valid, and the choice depends on your preference and the organization of your Terraform code. The main point is that provisioners are associated with a specific resource and are configured within the resource block or at the same level in the code.

upvoted 3 times

✉️  **Ramdi1** 7 months, 3 weeks ago

Selected Answer: B

the question is stated provisioner must be. the key word being Must. They do not have to be so I voted B

upvoted 3 times

✉️  **Ni33** 1 year, 2 months ago

Selected Answer: A

A is the correct answer

upvoted 1 times

✉️  **Bluemoon22** 1 year, 2 months ago

A: true.

for instance,

```
resource "aws_instance" "testInstance" {
  ami = "${var.instance_ami}"
  instance_type = "${var.instance_type}"
  subnet_id = "${aws_subnet.subnet_public.id}"
  vpc_security_group_ids = ["${aws_security_group.sg_22.id}"]
  key_name = "${aws_key_pair.ec2key.key_name}"
  tags {
    "Environment" = "${var.environment_tag}"
  }
  provisioner "local-exec" {
    command = "echo ${aws_instance.testInstance.public_ip} >> public_ip.txt"
  }
}
```

upvoted 2 times

✉️  **SaadKhamis** 1 year, 3 months ago

Selected Answer: B

Examples given in "<https://developer.hashicorp.com/terraform/language/resources/provisioners/connection>" are not inside a resource configuration block.

```
provisioner "file" {
  source = "conf/myapp.conf"
  destination = "/etc/myapp.conf"

  connection {
    type = "ssh"
    user = "root"
    password = "${var.root_password}"
    host = "${var.host}"
  }
}
```

upvoted 2 times

✉️  **halfway** 1 year, 1 month ago

That's only a code snippet, not the full configuration, though.

upvoted 4 times

✉️  **Power123** 1 year, 3 months ago

True. Ans is A

upvoted 1 times

✉️  **vyhak** 1 year, 5 months ago

A : True

upvoted 1 times

✉️  **kartikjena31** 1 year, 6 months ago

A- True

upvoted 1 times

✉️  **Bere** 1 year, 6 months ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/language/resources/provisioners/syntax>

Provisioners are used to execute scripts on a local or remote machine as part of resource creation or destruction. Provisioners can be used to bootstrap a resource, cleanup before destroy, run configuration management, etc.

How to use Provisioners

Note: Provisioners should only be used as a last resort. For most common situations there are better alternatives. For more information, see the sections above.

If you are certain that provisioners are the best way to solve your problem after considering the advice in the sections above, you can add a provisioner block inside the resource block of a compute instance.

```
resource "aws_instance" "web" {  
# ...  
  
provisioner "local-exec" {  
command = "echo The server's IP address is ${self.private_ip}"  
}  
}  
  
upvoted 3 times
```

✉️  **chimons** 1 year, 6 months ago

Selected Answer: A

A.

If you need to run provisioners that aren't directly associated with a specific resource, you can associate them with a null_resource.

upvoted 1 times

✉️  **dv00thay** 1 year, 7 months ago

Should be B

```
# Terraform Block  
terraform {  
required_version = ">= 1.0.0"  
required_providers {  
azurerm = {  
source = "hashicorp/azurerm"  
version = ">= 2.0"  
}  
random = {  
source = "hashicorp/random"  
version = ">= 3.0"  
}  
}  
}  
  
# Provider Block  
provider "azurerm" {  
features {}  
}
```

upvoted 1 times

✉️  **VincentMenzel** 1 year, 6 months ago

provisioner != provider

upvoted 6 times

✉️  **ziancom** 1 year, 5 months ago

LOL this is funny

upvoted 2 times

✉️  **faltu1985** 1 year, 9 months ago

Selected Answer: A

I think ans is A

upvoted 4 times

Terraform can run on Windows or Linux, but it requires a Server version of the Windows operating system.

- A. True
- B. False

Correct Answer: B

Community vote distribution

B (100%)

 **softarts** Highly Voted 2 years, 2 months ago

choose B
upvoted 16 times

 **javabadillo** Highly Voted 2 years ago

Selected Answer: B
B <https://www.terraform.io/downloads>
upvoted 5 times

 **tfdestroy** Most Recent 11 months, 3 weeks ago

Selected Answer: B
Terraform is platform-agnostic and it can run on many different systems, including both Windows and Linux. It does not require a Server version of the Windows operating system. You can run it on a standard Windows 10 installation, for example. The only requirement is that you have the Terraform binary installed and that it's in your system's PATH.
upvoted 4 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B
B is correct
upvoted 1 times

 **Bluemoon22** 1 year, 2 months ago

B, false
upvoted 1 times

 **Power123** 1 year, 3 months ago

B is the correct ans
upvoted 1 times

 **Eltooth** 2 years ago

Selected Answer: B
B is correct answer : false.
upvoted 4 times

What does the default "local" Terraform backend store?

- A. tfplan files
- B. Terraform binary
- C. Provider plugins

D. State file

Correct Answer: D

The local backend stores state on the local filesystem, locks that state using system APIs, and performs operations locally.

Reference:

<https://www.terraform.io/docs/language/settings/backends/local.html>

Community vote distribution

D (100%)

 **softarts** Highly Voted 2 years, 2 months ago

choose D

upvoted 12 times

 **Eltooth** Highly Voted 2 years ago

Selected Answer: D

D is correct answer.

<https://www.terraform.io/language/settings/backends/local>

<https://www.terraform.io/language/state>

upvoted 8 times

 **Bluemoon22** Most Recent 1 year, 2 months ago

D, state files

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: D

D. State file.

The default "local" Terraform backend stores the state file on the local disk of the machine running Terraform. The state file contains information about the resources managed by Terraform, such as their current state and any dependencies between them. When running Terraform commands, such as `terraform plan` or `terraform apply`, Terraform reads the state file to determine the current state of the resources and what changes need to be made to reach the desired state.

upvoted 4 times

 **Power123** 1 year, 3 months ago

D is correct

upvoted 1 times

 **vyhak** 1 year, 5 months ago

Selected Answer: D

D is the answer

upvoted 1 times

 **asat_chil** 1 year, 6 months ago

Answer: D

upvoted 1 times

 **Bere** 1 year, 6 months ago

Selected Answer: D

<https://developer.hashicorp.com/terraform/language/state>

Terraform must store state about your managed infrastructure and configuration. This state is used by Terraform to map real world resources to your configuration, keep track of metadata, and to improve performance for large infrastructures.

This state is stored by default in a local file named "terrafrom.tfstate", but it can also be stored remotely, which works better in a team environment.

upvoted 3 times

You have multiple team members collaborating on infrastructure as code (IaC) using Terraform, and want to apply formatting standards for readability.

How can you format Terraform HCL (HashiCorp Configuration Language) code according to standard Terraform style convention?

- A. Run the `terraform fmt` command during the code linting phase of your CI/CD process
- B. Designate one person in each team to review and format everyone's code
- C. Manually apply two spaces indentation and align equal sign "=" characters in every Terraform file (*.tf)
- D. Write a shell script to transform Terraform files using tools such as AWK, Python, and sed

Correct Answer: C

- ⇒ Indent two spaces for each nesting level.
- ⇒ When multiple arguments with single-line values appear on consecutive lines at the same nesting level, align their equals signs.

Reference:

<https://www.terraform.io/docs/language/syntax/style.html>

Community vote distribution

A (100%)

✉️  **vitasac** Highly Voted 2 years, 2 months ago

Selected Answer: A

Yes Answer A

upvoted 33 times

✉️  **calebvar** Highly Voted 2 years, 2 months ago

correct answer is A

upvoted 11 times

✉️  **nickyop** Most Recent 4 months, 1 week ago

Okay, as obvious as the option A is the correct answer, here's one question. What happens if such questions arise in the terraform exam when they have the wrong answers saved as the right ones? For example, in the above question, if we choose the most voted option in the exam, we lose 1 mark just because the right answer is C according to the system?? And FYI, I have already undergone one attempt for the exam which was certain that I would pass because it went pretty well but I failed by 2%. And I know for sure it is only because of such contradictory questions! @Hashicorp please settle such questions once and for all!

upvoted 2 times

✉️  **AkaAka4** 2 months, 3 weeks ago

Why are you even tagging Hashicorp here... do you think they will see your comment 😂

upvoted 3 times

✉️  **ghostGuiggs** 8 months, 2 weeks ago

Selected Answer: A

A is the answer

upvoted 2 times

 **Tyler2023** 8 months, 4 weeks ago

The answer C is correct

You can manually format your TF configuration to follow the Style Conventions

Option A doesn't make any sense, why you run the terraform fmt in CI/CI process? that will only format the copy of the configuration in the remote agent that is running your configuration files, it doesn't modify your source codes in version control system.

Option C followed the Terraform style convention, but it only do it manually

<https://developer.hashicorp.com/terraform/language/syntax/style>

upvoted 1 times

 **nickyop** 4 months, 1 week ago

Referring the link you have shared, I think you missed a note where they have mentioned "You can enforce these conventions automatically running terraform fmt."

upvoted 2 times

 **Clapton79** 8 months, 1 week ago

Pull Request is a CI/CD process and it can format code and it does modify your version control.

upvoted 1 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: A

A for sure

upvoted 1 times

 **vc1011** 1 year ago

Selected Answer: A

It's very evident A

upvoted 1 times

 **Awadh** 1 year ago

A is the correct answer, who is writing the answers

upvoted 4 times

 **Shane_C** 1 year ago

Selected Answer: A

Now they have to be joking.

The only reasonable answer is A.

Everything else is just nonsense

upvoted 3 times

 **nickyop** 4 months, 1 week ago

Yes, thank you!

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

A is correct

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

A is the correct answer

upvoted 1 times

 **Bluemoon22** 1 year, 2 months ago

A, by using terraform fmt command

upvoted 1 times

 **connecttozee** 1 year, 3 months ago

A is correct

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: A

A. Run the terraform fmt command during the code linting phase of your CI/CD process.

Terraform provides a command called terraform fmt that can be used to automatically format Terraform HCL code according to the standard Terraform style convention. Running this command on your codebase during the code linting phase of your CI/CD process can ensure that your code is formatted consistently across your team and conforms to the standard Terraform style convention.

upvoted 6 times

 **Power123** 1 year, 3 months ago

A is correct

upvoted 1 times

 **hasanuckun** 1 year, 4 months ago

answer is a

upvoted 1 times

 **alexandroe** 1 year, 4 months ago

Selected Answer: A

FMT helped de code

upvoted 1 times

Question #36

Topic 1

What value does the Terraform Cloud/Terraform Enterprise private module registry provide over the public Terraform Module Registry?

- A. The ability to share modules with public Terraform users and members of Terraform Enterprise Organizations
- B. The ability to tag modules by version or release
- C. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations
- D. The ability to share modules publicly with any user of Terraform

Correct Answer: D

Terraform Registry is an index of modules shared publicly using this protocol. This public registry is the easiest way to get started with Terraform and find modules created by others in the community.

Reference:

<https://www.terraform.io/docs/language/modules/sources.html>

Community vote distribution

C (100%)

 **vitasac** Highly Voted 2 years, 2 months ago

Selected Answer: C

for sure C

upvoted 17 times

 **tipzzz** Highly Voted 2 years, 2 months ago

Selected Answer: C

Terraform Cloud's private registry works similarly to the public Terraform Registry and helps you share Terraform providers and Terraform modules across your organization. It includes support for versioning and a searchable list of available providers and modules.

upvoted 10 times

 **hoangphan** Most Recent 5 months, 2 weeks ago

Selected Answer: C

of course C

upvoted 1 times

👤 **samimshaikh** 6 months, 2 weeks ago

Selected Answer: C

the question has the word private registry "over" the public

C. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations

The Terraform Cloud/Terraform Enterprise private module registry provides the ability to restrict modules to members of specific Terraform Cloud or Enterprise organizations. This allows organizations to control access to and usage of modules, keeping them private within the organization. Option C correctly describes this capability.

upvoted 1 times

👤 **Yhorm** 8 months, 2 weeks ago

sometime I wonder whether whomever picked the 'correct' answers just picked an alternative at random

upvoted 3 times

👤 **Bere** 11 months, 3 weeks ago

Selected Answer: C

The private module registry in Terraform Cloud and Terraform Enterprise is a way to distribute Terraform modules within your organization. The public Terraform Module Registry, on the other hand, is open to everyone.

Here is an example of how you might use a module from a private module registry:

```
module "vpc" {
  source = "app.terraform.io/example_corp/vpc/aws"
  version = "1.0.0"
```

```
// ...other arguments...
}
```

Here is an example of how you might use a module from the public Terraform Module Registry:

```
module "vpc" {
  source = "terraform-aws-modules/vpc/aws"
  version = "2.77.0"
```

```
// ...other arguments...
}
```

upvoted 5 times

👤 **Shane_C** 1 year ago

Selected Answer: C

Come on guys, it's C

upvoted 2 times

👤 **Ni33** 1 year, 2 months ago

Selected Answer: C

C is the correct answer

upvoted 1 times

👤 **Faaizz** 1 year, 3 months ago

Selected Answer: C

Only C makes sense here

upvoted 1 times

👤 **camps** 1 year, 3 months ago

Selected Answer: C

C. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations.

The private module registry in Terraform Cloud/Terraform Enterprise provides an additional level of control and security over the public Terraform Module Registry. Unlike the public registry, the private registry allows organizations to restrict module access to only members of their Terraform Cloud or Enterprise organization. This ensures that sensitive infrastructure code is not accidentally or intentionally shared with unauthorized users.

upvoted 6 times

 **camps** 1 year, 3 months ago

Selected Answer: C

C. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations.

The Terraform Module Registry is a public repository of Terraform modules that can be used by anyone using Terraform. The Terraform Cloud/Terraform Enterprise private module registry provides additional functionality for organizations that want to create and manage their own private modules.

The private module registry provides several benefits over the public Terraform Module Registry, including:

The ability to restrict modules to members of Terraform Cloud or Enterprise organizations: This allows organizations to control who has access to their private modules and prevent unauthorized access.

The ability to tag modules by version or release: This makes it easy to manage and track changes to modules over time.

The ability to manage module dependencies: This allows organizations to manage and version the dependencies of their private modules.

Integration with Terraform Cloud or Enterprise workspaces: This allows organizations to seamlessly use their private modules in their Terraform Cloud or Enterprise workspaces.

upvoted 3 times

 **Power123** 1 year, 3 months ago

Answer is C

upvoted 1 times

 **alexandroe** 1 year, 4 months ago

Selected Answer: C

C is correct

upvoted 1 times

 **Bere** 1 year, 6 months ago

Selected Answer: C

<https://developer.hashicorp.com/terraform/cloud-docs/registry>

Private Registry

Terraform Cloud's private registry works similarly to the public Terraform Registry and helps you share Terraform providers and Terraform modules across your organization. It includes support for versioning and a searchable list of available providers and modules.

<https://developer.hashicorp.com/terraform/cloud-docs/registry#private-providers-and-modules>

Private Providers and Modules

Private providers and private modules are hosted on an organization's private registry and are only available to members of that organization. Terraform Enterprise, private modules are also available to other organizations that are configured to share modules with that organization.

upvoted 3 times

 **mendelthegreat** 1 year, 8 months ago

C is the correct answer DUH

upvoted 1 times

 **legendary7** 1 year, 9 months ago

C - is the correct answer.

D- is simply the advantage of Public registry over Private registry. It is the opposite of what the question asks

upvoted 3 times

 **yuvifose** 1 year, 12 months ago

Selected Answer: C

The answer should be C

upvoted 1 times

- A. Sources all providers present in the configuration and ensures they are downloaded and available locally
- B. Connects to the backend
- C. Sources any modules and copies the configuration locally
- D. Validates all required variables are present

Correct Answer: D

Reference:

<https://www.terraform.io/docs/cli/commands/init.html>

Usage

Usage: `terraform init [options]`

This command performs several different initialization steps in order to prepare the current working directory for use with Terraform. More details on these are in the sections below, but in most cases it is not necessary to worry about these individual steps.

This command is always safe to run multiple times, to bring the working directory up to date with changes in the configuration. Though subsequent runs may give errors, this command will never delete your existing configuration or state.

Community vote distribution

D (96%)

4%

 **elvancedonzy** Highly Voted 2 years, 1 month ago

Selected Answer: D

D is correct

upvoted 12 times

 **DarrylNg** Most Recent 2 months, 3 weeks ago

Selected Answer: D

D is correct. You can test it out in a sample tf file. Init command goes through even if some variable that another resource is referencing is missing.

upvoted 1 times

 **SilentH** 3 months ago

Selected Answer: D

Finally ExamTopics got 1 right!

upvoted 4 times

 **Bere** 11 months, 3 weeks ago

Selected Answer: D

terraform init performs several initialization steps to prepare your Terraform working directory to be used with Terraform commands. However, does not validate whether all required variables are present in the configuration or provided through another method. That validation happens during the terraform plan or terraform apply stage.

terraform init does:

1. Download and installs the necessary provider plugins.
2. Setup backend for storing state.
3. Download and install modules

upvoted 2 times

 **Ni33** 1 year, 2 months ago

I think A is the correct answer.

upvoted 3 times

 **Bolgarwow** 3 months, 2 weeks ago

terraform init not perform

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

A is the correct answer.

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: D

D. Validates all required variables are present.

terraform init is a command that initializes a new or existing Terraform configuration. When you run terraform init, Terraform performs several tasks to set up the configuration for use, including:

Sources all providers present in the configuration and ensures they are downloaded and available locally

Connects to the backend, if one is configured, and performs any necessary setup steps

Sources any modules referenced in the configuration and copies their contents locally

Initializes the backend configuration and performs any necessary setup steps

upvoted 2 times

 **Power123** 1 year, 3 months ago

Ans is C. init doesn't validate all variables are present

upvoted 2 times

 **bwahdi** 1 year, 7 months ago

why isn't it A?

upvoted 2 times

 **FarziWaliMarzi** 1 year, 6 months ago

focus on the key word "not"

upvoted 2 times

 **Burakko** 1 year, 10 months ago

Selected Answer: D

terraform plan or apply does "validates all required variables are present"

upvoted 1 times

 **flaviu888** 2 years ago

yes, should be D

upvoted 2 times

 **Eltooth** 2 years ago

Selected Answer: D

D is correct answer.

upvoted 2 times

 **Ahmad_Terraform** 2 years ago

yes INIT does not validate variables

upvoted 1 times

 **habros** 2 years, 1 month ago

Looks like C to me

upvoted 2 times

 **Oskar_Madin** 2 years ago

go and learn first no one need your looks like

upvoted 7 times

Question #38

Topic 1

You have declared a variable called var.list which is a list of objects that all have an attribute id.
Which options will produce a list of the IDs? (Choose two.)

- A. { for o in var.list : o => o.id }
- B. var.list[*].id
- C. [var.list[*].id]
- D. [for o in var.list : o.id]

Correct Answer: AB

Community vote distribution

BD (90%)

10%

 **bigboi23** Highly Voted 2 years, 1 month ago

Selected Answer: BD

<https://www.terraform.io/language/expressions/splat>

A splat expression provides a more concise way to express a common operation that could otherwise be performed with a for expression.
upvoted 23 times

✉️  **Bere**  11 months, 3 weeks ago

Selected Answer: BD

Here's an example:

Assume you have the following variable declaration:

```
variable "users" {
  default = [
    {
      id = "id1"
      name = "name1"
    },
    {
      id = "id2"
      name = "name2"
    },
    {
      id = "id3"
      name = "name3"
    }
  ]
}
```

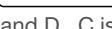
You can retrieve the list of IDs in your Terraform configuration using either of these options:

```
output "users_splat" {
  value = var.users["*"].id
}

output "users_for" {
  value = [for user in var.users : user.id]
}
```

Both these outputs will produce the same list of IDs: ["id1", "id2", "id3"].

upvoted 10 times

✉️  **vibzr2023**  3 months, 2 weeks ago

Apart from B and D , C is also correct

```
> [for o in var.list : o.id]
[
  "1",
  "2",
  "3",
]
```

upvoted 2 times

✉️  **zimomar** 5 months, 3 weeks ago

Selected Answer: BD

BD for sure

upvoted 1 times

✉️  **ndiichie** 11 months, 3 weeks ago

Answer is BD.

<https://developer.hashicorp.com/terraform/language/expressions/splat>

upvoted 2 times

✉️  **srajvanshi** 12 months ago

<https://developer.hashicorp.com/terraform/language/expressions/splat>

B and D

upvoted 1 times

✉️  **krishna2802** 1 year ago

Selected Answer: BD

<https://www.terraform.io/language/expressions/splat>

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: AB

A and B is the correct answer

upvoted 3 times

 **AzRNoob** 1 year, 3 months ago

the correct options that will produce a list of the IDs are B and D:

Option B, var.list[*].id, uses the splat operator [*] to iterate over all elements of the var.list list and then accesses the id attribute of each object. The result is a list of all the id values.

Option D, [for o in var.list : o.id], uses a list comprehension to iterate over each object in the var.list list and create a new list that contains only the id attribute of each object.

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: BD

B. var.list[*].id and D. [for o in var.list : o.id].

To produce a list of IDs from a list of objects with an id attribute, you can use either of the following options:

var.list[*].id: This uses the [*] wildcard to select all elements of the var.list list, and the .id syntax to select the id attribute of each element. This produces a list of IDs.

[for o in var.list : o.id]: This uses a for expression to iterate over each element of var.list, selecting the id attribute of each element. This will produce a list of IDs.

upvoted 2 times

 **Power123** 1 year, 3 months ago

The splat and interpolation. Ans is B & D

upvoted 1 times

 **thor7** 1 year, 3 months ago

Selected Answer: BD

B and D are correct, checked them.

Reference: <https://developer.hashicorp.com/terraform/language/expressions/for>

upvoted 1 times

 **Mal_8** 1 year, 4 months ago

Selected Answer: BD

BD. Tried each expression

upvoted 1 times

 **sahara99** 1 year, 5 months ago

Selected Answer: BD

A is wrong as the question asked which one creates a "list"

upvoted 2 times

 **Tanacet** 1 year, 5 months ago

Selected Answer: BD

The splat and interpolation-style

upvoted 1 times

 **Mohammed52** 1 year, 5 months ago

Selected Answer: BD

B and D is correct

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

Selected Answer: BD

B and D is correct

upvoted 1 times

Which argument(s) is (are) required when declaring a Terraform variable?

- A. type
- B. default
- C. description
- D. All of the above
- E. None of the above

Correct Answer: B

The variable declaration can also include a default argument.

Reference:

<https://www.terraform.io/docs/language/values/variables.html>

Community vote distribution

E (94%)

6%

✉️  Egger1992 Highly Voted 2 years, 2 months ago

Selected Answer: E

The type argument in a variable block allows you to restrict the type of value that will be accepted as the value for a variable. If no type constraint is set then a value of any type is accepted.

Source: <https://www.terraform.io/language/values/variables>

upvoted 30 times

✉️  AWS_PT 1 year, 8 months ago

it is not mandatory though! E is correct

upvoted 2 times

✉️  Shane_C 1 year ago

He selected E as his answer

upvoted 2 times

✉️  gcpz Highly Voted 2 years, 2 months ago

Correct is E

Terraform CLI defines the following OPTIONAL arguments for variable declarations:

default - A default value which then makes the variable optional.

type - This argument specifies what value types are accepted for the variable.

description - This specifies the input variable's documentation.

validation - A block to define validation rules, usually in addition to type constraints.

sensitive - Limits Terraform UI output when the variable is used in configuration.

nullable - Specify if the variable can be null within the module.

upvoted 17 times

✉️  Molly1994 Most Recent 1 month, 1 week ago

they are all optional arguments for variable declarations. so E is correct none of them are required

upvoted 1 times

✉️  **samimshaikh** 6 months, 2 weeks ago

Selected Answer: E

we can define variables without type, default, and description.

default argument will require and value when terraform apply is executed. So in the context of this question is it required and answer is not

```
variable "prefix" {  
}
```

```
variable "location" {  
}
```

upvoted 1 times

✉️  **ghostGuiggs** 8 months, 2 weeks ago

Selected Answer: E

E is the answer

upvoted 1 times

✉️  **Tyler2023** 8 months, 4 weeks ago

Correct, the answer is E

<https://developer.hashicorp.com/terraform/language/values/variables#disallowing-null-input-values>

upvoted 1 times

✉️  **Bere** 11 months, 3 weeks ago

Selected Answer: E

When declaring a variable in Terraform, no arguments are required. However, there are optional arguments you can include for clarity and readability:

description: This is a string that describes the purpose of the variable.

default: This is the default value that Terraform will use if no other value is provided.

type: This constrains the type of value the variable will accept, such as string, list, map, number, or bool.

Here's an example of a variable declaration with no arguments:

```
variable "example" {  
}
```

And here's an example with all optional arguments included:

```
variable "example" {  
  description = "An example variable used for demonstration purposes"  
  default = "Hello, World!"  
  type = string  
}
```

upvoted 1 times

✉️  **mirekbehan** 1 year ago

Selected Answer: E

Yes E is correct.

upvoted 1 times

✉️  **Shane_C** 1 year ago

E is correct

upvoted 1 times

✉️  **vj_dhaksh** 1 year, 1 month ago

Arguments

Terraform CLI defines the following optional arguments for variable declarations:

default - A default value which then makes the variable optional.

type - This argument specifies what value types are accepted for the variable.

description - This specifies the input variable's documentation.

validation - A block to define validation rules, usually in addition to type constraints.

sensitive - Limits Terraform UI output when the variable is used in configuration.

nullable - Specify if the variable can be null within the module.

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: E

E is the correct answer

upvoted 1 times

 **AzRNoob** 1 year, 3 months ago

E is correct.

It is possible to declare a variable without any of the given arguments (type, default, description). In fact, a variable declaration only requires a name and an optional type constraint.

So the correct answer is E. None of the above.

upvoted 2 times

 **Faaizz** 1 year, 3 months ago

Selected Answer: E

Definitely E

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: E

None of the above arguments are required when declaring a Terraform variable. However, at least one of the following arguments is required:

type: Specifies the data type of the variable. Valid types include string, number, bool, list, map, and object.

default: Specifies the default value of the variable. If no default value is specified, the variable is considered required and must be set by the user when running Terraform.

description: Provides a description of the variable and its intended use. This is optional but recommended to make the purpose of the variable clear to users.

In addition to these arguments, there are several other optional arguments that can be used when declaring a Terraform variable, including validation, sensitive, and allowed_values.

upvoted 1 times

 **Power123** 1 year, 3 months ago

Answer is E. All are optional

upvoted 1 times

 **sahara99** 1 year, 5 months ago

Selected Answer: E

none of the options is mandatory - they are all optional

upvoted 1 times

 **vikramv1r** 1 year, 7 months ago

Answer is E.

we can declare the variable as

variable "varname" {}

type,default and description are optional..

upvoted 1 times

Question #40

Topic 1

When using a module block to reference a module stored on the public Terraform Module Registry such as:

```
module "consul" {
    source = "hashicorp/consul/aws"
}
```

How do you specify version 1.0.0?

- A. Modules stored on the public Terraform Module Registry do not support versioning
- B. Append ?ref=v1.0.0 argument to the source path
- C. Add version = "1.0.0" attribute to module block
- D. Nothing modules stored on the public Terraform Module Registry always default to version 1.0.0

Correct Answer: C*Community vote distribution*

C (100%)

  **amrith501** Highly Voted 2 years, 1 month ago**Selected Answer: C**

C correct answer

Version

When using modules installed from a module registry, we recommend explicitly constraining the acceptable version numbers to avoid unexpected or unwanted changes.

Use the version argument in the module block to specify versions:

```
module "consul" {
  source = "hashicorp/consul/aws"
  version = "0.0.5"

  servers = 3
}
upvoted 17 times
```

  **enook** Most Recent 6 months, 2 weeks ago**Selected Answer: C**

CCCCCC

upvoted 1 times

  **Bere** 11 months, 3 weeks ago**Selected Answer: C**

Example:

```
module "consul" {
  source = "hashicorp/consul/aws"
  version = "1.0.0"
  // other necessary arguments...
}
upvoted 2 times
```

  **foreverlearner** 1 year ago**Selected Answer: C**

Although please note that version is now deprecated in favour of the required_providers block
<https://developer.hashicorp.com/terraform/language/providers/configuration#version-deprecated>

upvoted 2 times

  **foreverlearner** 1 year ago

Ok sorry my bad this is a module not a provider so that's not relevant (but still good to know though :))

Answer is still C

upvoted 1 times

  **Ni33** 1 year, 2 months ago**Selected Answer: C**

C is the correct answer. It is used to lock down version of the modules in production grade infrastructure templates.

upvoted 1 times

  **Power123** 1 year, 3 months ago

C is correct

upvoted 1 times

✉️  **sahara99** 1 year, 5 months ago

Selected Answer: C

example from HashiCorp:
module "consul" {
source = "hashicorp/consul/aws"
version = "0.0.5"

servers = 3
}

upvoted 2 times

✉️  **aleksand41** 1 year, 9 months ago

Selected Answer: C

C is correct

upvoted 1 times

✉️  **Eltooth** 2 years ago

Selected Answer: C

C is correct answer.

The version argument accepts a version constraint string. Terraform will use the newest installed version of the module that meets the constraint if no acceptable versions are installed, it will download the newest version that meets the constraint.

Version constraints are supported only for modules installed from a module registry, such as the public Terraform Registry or Terraform Cloud's private module registry. Other module sources can provide their own versioning mechanisms within the source string itself, or might not support versions at all. In particular, modules sourced from local file paths do not support version; since they're loaded from the same source repository they always share the same version as their caller.

<https://www.terraform.io/language/modules/syntax#version>

upvoted 3 times

✉️  **elvancedonzy** 2 years, 1 month ago

Selected Answer: C

C is correct

<https://www.terraform.io/language/modules/syntax>

upvoted 3 times

Question #41

Topic 1

What features does the hosted service Terraform Cloud provide? (Choose two.)

- A. Automated infrastructure deployment visualization
- B. Automatic backups
- C. Remote state storage
- D. A web-based user interface (UI)

Correct Answer: BC

Reference:

<https://www.terraform.io/docs/enterprise/admin/automated-recovery.html> <https://www.terraform.io/docs/language/state/remote.html>

Remote State

JUMP TO SECTION ▾

By default, Terraform stores state locally in a file named `terraform.tfstate`. When working with Terraform in a team, use of a local file makes Terraform usage complicated because each user must make sure they always have the latest state data before running Terraform and make sure that nobody else runs Terraform at the same time.

With *remote state*, Terraform writes the state data to a remote data store, which can then be shared between all members of a team. Terraform supports storing state in [Terraform Cloud](#), [HashiCorp Consul](#), Amazon S3, Azure Blob Storage, Google Cloud Storage, Alibaba Cloud OSS, and more.

Remote state is implemented by a [backend](#) or by Terraform Cloud, both of which you can configure in your configuration's root module.

Community vote distribution

CD (100%)

✉️  **cytron**  2 years, 1 month ago

Selected Answer: CD

CD because there is web UI

upvoted 20 times

✉️  **lmrab**  1 year, 2 months ago

C. Remote state storage

D. A web-based user interface (UI)

Terraform Cloud is a hosted service provided by HashiCorp that allows users to centrally manage and automate their infrastructure deployments. It provides a web-based user interface (UI) that allows users to manage their infrastructure resources, configurations, and state files, as well as collaborate with team members. Terraform Cloud also provides remote state storage, which is a critical feature of Terraform that allows users to store the state of their infrastructure resources in a central location, making it easier to manage and update their deployments. However, Terraform Cloud does not provide automated infrastructure deployment visualization or automatic backups as features.

upvoted 9 times

✉️  **Power123**  1 year, 3 months ago

C and D

upvoted 1 times

✉️  **AzDev937** 1 year, 6 months ago

Selected Answer AD.

A. Automated infrastructure deployment visualization

D. A web-based user interface (UI)

Terraform Cloud does not provide automatic backups or remote state storage as features. Option B: Automatic backups is not a feature of Terraform Cloud. Option C: Remote state storage is a feature of Terraform Enterprise, which is a separate product from Terraform Cloud.

upvoted 1 times

✉️  **lo_ol** 1 year, 5 months ago

Terraform Cloud is a commercial SaaS product developed by HashiCorp. Many of its features are free for small teams, including remote state storage, remote runs, and VCS connections. We also offer paid plans for larger teams that include additional collaboration and governance features

<https://developer.hashicorp.com/terraform/cloud-docs/overview>

upvoted 4 times

 **InformationOverload** 1 year, 6 months ago

Selected Answer: CD

C & D is correct

upvoted 2 times

 **adouban** 1 year, 7 months ago

Selected Answer: CD

C,D relevant to me

upvoted 2 times

 **cjig** 1 year, 11 months ago

Selected Answer: CD

C and D are correct answers, there is no auto backup in Terraform cloud.

<https://www.terraform.io/enterprise/admin/infrastructure/backup-restore>

upvoted 7 times

 **Eltooth** 2 years ago

Selected Answer: CD

C and D are correct answers.

upvoted 3 times

 **Ahmad_Terraform** 2 years ago

BCD looks fitting

upvoted 1 times

 **Raynman727** 2 years, 1 month ago

Is this not CD? There is definitely a Web UI for Terraform Cloud

upvoted 2 times

 **pj2001** 2 years, 1 month ago

should be CD?

upvoted 2 times

Question #42

Topic 1

Where does the Terraform local backend store its state?

- A. In the /tmp directory
- B. In the terraform file
- C. In the terraform.tfstate file
- D. In the user's terraform.state file

Correct Answer: C

The local backend stores state on the local filesystem, locks that state using system APIs, and performs operations locally.

Reference:

<https://www.terraform.io/docs/language/settings/backends/local.html>

Community vote distribution

C (100%)

 **mav3r1ck** Highly Voted 1 year, 10 months ago

C.

<https://www.terraform.io/language/settings/backends/local>

upvoted 7 times

 **090200f** Most Recent 5 days ago

terraform.tfstate , C is the correct ans

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: C

C is the correct answer

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: C

C is the correct answer.

upvoted 1 times

 **Power123** 1 year, 3 months ago

C. terraform.tfstate

upvoted 1 times

 **gekkehenk** 1 year, 4 months ago

Selected Answer: C

The local backend stores state on the local filesystem, locks that state using system APIs, and performs operations locally.

<https://developer.hashicorp.com/terraform/language/settings/backends/local>

upvoted 3 times

 **samkobadev** 1 year, 4 months ago

C. .tfstate file

upvoted 1 times

 **kennynelcon** 1 year, 6 months ago

Selected Answer: C

C state.tf file

upvoted 1 times

 **root_i_am** 1 year, 10 months ago

Selected Answer: C

The state is stored in the terraform.tfstate file in the root directory of the terraform project

upvoted 1 times

- A. A Terraform provider
- B. Environment variables
- C. A -var flag
- D. secure string

Correct Answer: C

Reference:

<https://secrethub.io/blog/secret-management-for-terraform/>

Community vote distribution

D (76%)

10%

10%

✉️  **hip9k** Highly Voted 1 year, 10 months ago

Selected Answer: D

It's D

We can use providers to supply variable values (vault for example).
 We can provide input variable value in parameter for apply command.
 We can use environment variables.
 HashiCorp is not mentioning anything about secure strings.

Reference:

<https://www.terraform.io/language/values/variables>

upvoted 25 times

✉️  **[Removed]** 1 year ago

Terraform does not have a built-in concept of a "secure string". This means that you cannot use the `secure_string` keyword to define a secret in your Terraform configuration file.

Link below recommends the three options.

A. e.g. Vault
 B. e.g. `export TF_VAR_db_username=admin TF_VAR_db_password=adifferentpassword`
 C. `-var-file="secret.tfvars"`
<https://developer.hashicorp.com/terraform/tutorials/configuration-language/sensitive-variables>

upvoted 2 times

✉️  **stalk98** Highly Voted 2 years, 1 month ago

i think D

upvoted 8 times

✉️  **CryptoShade** Most Recent 2 months, 4 weeks ago

Answer is: A. Terraform provider
 It says: to Hide secrets and not include secrets.
 Here's why the other options are suitable for hiding secrets:

B. Environment variables: Environment variables store sensitive information outside of Terraform code, and Terraform can access them during execution.
 C. A -var flag: The -var flag allows passing secrets as command-line arguments when running `terraform apply` or other commands. These arguments aren't stored in the configuration files.
 D. Secure string: Some Terraform providers (like AWS) offer functionality to store secrets securely within the provider itself (e.g., AWS Secrets Manager). This keeps them out of the configuration files.

upvoted 2 times

✉️  **Felienator** 3 months ago

swear to god these questions are worded so fking poorly
 upvoted 1 times

✉️  **Bolgarwow** 3 months, 2 weeks ago

can not be used to keep secrets
 D - Secret String
 upvoted 1 times

 **vibzr2023** 3 months, 2 weeks ago

D is correct.. In Terraform, the term "secure string" isn't a specific built-in type or feature by that name. However, the concept of treating certain strings as "secure" or sensitive is indeed present in Terraform, particularly through the use of the sensitive attribute for variables and outputs. When we refer to a "secure string" in the context of Terraform, it's generally about handling sensitive values such as passwords, secret keys, or any confidential data that should not be exposed in logs or CLI output.

Here's how you can declare a variable as sensitive:

```
variable "api_secret_key" {
  type = string
  sensitive = true
}
```

upvoted 1 times

 **imkhan** 8 months, 1 week ago

I will go for A.

All other options are to keep secrets out of Terraform configuration files, you typically use environment variables, a -var flag, or secure string variables.

upvoted 2 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: D

D, "secure string"

upvoted 1 times

 **Pradh** 9 months, 3 weeks ago

C is the answer

upvoted 1 times

 **Spandrop** 10 months, 2 weeks ago

Bad answers for this question.

Definitely you cannot use a terraform provider to keep secrets out of your terraform configuration.

Even if you use Vault, you must provide the Vault itself secrets and/or you save to a file, in an environment variable, or within the provider itself. So "A" is wrong.

The issue is that "D" is also wrong.

A and D should be the answers for this question in my opinion.

upvoted 2 times

 **BtotheJ** 11 months, 1 week ago

Selected Answer: D

D is correct because all other options can be used to keep secrets out of terraform config files

upvoted 1 times

 **Bere** 11 months, 2 weeks ago

Selected Answer: D

Answer is D.

A. Terraform Provider: You can use sensitive variables in Terraform Cloud (below link) or other secrets management solutions (e.g. AWS Secret Manager).

Sensitive variables / sensitive values is described here:

<https://developer.hashicorp.com/terraform/cloud-docs/workspaces/variables/managing-variables#sensitive-values>

B. Environment Variables: You can use environment variables. Terraform will read environment variables that start with TF_VAR_, followed by the name of a declared variable in your configuration.

C. -var flag: You can use the -var command line flag. This is useful for setting sensitive data that should not be stored in your configuration. e.g. `terraform apply -var 'db_password=My$ecretP@ssw0rd'`

D. "secure string" is not a valid option for keeping secrets out of Terraform configuration files. The term "secure string" is not a recognized or standard feature in Terraform.

upvoted 2 times

 **Jlee7** 1 year ago

Answer is A

A Terraform provider is a software library that allows Terraform to interact with a particular cloud provider or other infrastructure service. Terraform providers do not have the ability to store secrets, so they cannot be used to keep secrets out of Terraform configuration files.

upvoted 4 times

👤 March2023 1 year, 1 month ago

Selected Answer: D

Terraform does not have a built-in "secure string" option
upvoted 1 times

👤 milan92stankovic 1 year, 1 month ago

Selected Answer: D

It's D.
upvoted 1 times

👤 mememu 1 year, 2 months ago

A is incorrect,
A provider can also declare an attribute as sensitive, which will cause Terraform to hide it from regular output regardless of how you assign it a value.
Ref. <https://developer.hashicorp.com/terraform/language/values/variables>
upvoted 2 times

👤 Ni33 1 year, 2 months ago

Selected Answer: A

I think it is A. Provider has nothing to do with secret Management.
upvoted 1 times

Question #44

Topic 1

What is one disadvantage of using dynamic blocks in Terraform?

- A. They cannot be used to loop through a list of values
- B. Dynamic blocks can construct repeatable nested blocks
- C. They make configuration harder to read and understand
- D. Terraform will run more slowly

Correct Answer: A

Reference:

<https://github.com/hashicorp/terraform/issues/19291>

Community vote distribution

C (100%)

👤 ItaloVinodi Highly Voted 2 years, 2 months ago

Selected Answer: C

Looking at documentation "Overuse of dynamic blocks can make configuration hard to read and maintain"
upvoted 13 times

👤 Bere Highly Voted 11 months, 2 weeks ago

Selected Answer: C

Answer is C

They make configuration harder to read and understand.

While dynamic blocks in Terraform are powerful tools that can create multiple blocks based on complex input structures, their usage can also lead to configurations that are harder to read and understand. This is because the logic for creating the dynamic blocks is usually more complex and less straightforward than static block configurations.

upvoted 5 times

 **cafl7787** (Most Recent) 4 months, 3 weeks ago

For sure C.

upvoted 1 times

 **syam22587** 10 months, 4 weeks ago

Selected Answer: C

C is correct

upvoted 2 times

 **Mehkay** 11 months ago

Correct answer C

upvoted 1 times

 **srajvanshi** 12 months ago

<https://developer.hashicorp.com/terraform/language/expressions/dynamic-blocks>

Overuse of dynamic blocks can make configuration hard to read and maintain, so we recommend using them only when you need to hide details in order to build a clean user interface for a re-usable module. Always write nested blocks out literally where possible.

upvoted 4 times

 **Power123** 1 year, 3 months ago

Answer is C

upvoted 1 times

 **alexandroe** 1 year, 4 months ago

Selected Answer: C

correct answer

upvoted 1 times

 **sahara99** 1 year, 5 months ago

Selected Answer: C

reading dynamic blocks doc. --> C is the correct answer

upvoted 1 times

 **zaz_** 1 year, 6 months ago

C is the correct Option.

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

Selected Answer: C

C is correct! Dynamic blocks allow you to define infrastructure resources using variables, which can make it harder to see exactly what resources are being created. This can make it more difficult to troubleshoot issues and understand the overall architecture of your infrastructure.

upvoted 3 times

 **Atta33** 1 year, 7 months ago

They make configuration hard to read and understand.

upvoted 1 times

 **legendary7** 1 year, 9 months ago

C- is the correct answer

Ref: <https://www.terraform.io/language/expressions/dynamic-blocks#best-practices-for-dynamic-blocks>

upvoted 2 times

 **dnscloud02** 1 year, 9 months ago

Selected Answer: C

C is correct answer.

upvoted 1 times

 **yuvifose** 1 year, 12 months ago

Selected Answer: C

C is the correct answer. It's not recommended to abuse them because it hurts readability

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

They make configuration harder to read and understand

upvoted 1 times

 **trongod05** 2 years ago

C "Overuse of dynamic blocks can make configuration hard to read and maintain, so we recommend using them only when you need to hide details in order to build a clean user interface for a re-usable module. Always write nested blocks out literally where possible."

upvoted 4 times

Question #45

Topic 1

Only the user that generated a plan may apply it.

A. True

B. False

Correct Answer: B

Community vote distribution

B (96%)

4%

 **Jaro3000**  2 years, 1 month ago

I suppose they do it by purpose so we discuss it

upvoted 21 times

 **vibzr2023** 3 months, 2 weeks ago

you made my day :)

upvoted 1 times

 **kopper2019** 1 year, 11 months ago

this made me laugh :)

upvoted 4 times

 **yuvifose**  1 year, 12 months ago

B is correct, a plan can be stored as a file and another person can execute the plan file

upvoted 12 times

 **Bere**  11 months, 2 weeks ago

Selected Answer: B

For example, in a continuous deployment pipeline, one job might run terraform plan -out=plan.out to generate the plan, and this plan could be applied later in a separate job (potentially triggered manually for additional control) using terraform apply plan.out. These jobs might be run on different user accounts, or even on different machines.

As described here: https://developer.hashicorp.com/terraform/tutorials/automation/automate-terraform?utm_source=WEBSITE&utm_medium=WEB_IO&utm_offer=ARTICLE_PAGE&utm_content=DOCS

Steps 1, 2 and 4 can be carried out using the familiar Terraform CLI commands, with some additional options:

terrafrom init -input=false to initialize the working directory.

terrafrom plan -out=tfplan -input=false to create a plan and save it to the local file tfplan.

terrafrom apply -input=false tfplan to apply the plan stored in the file tfplan.

upvoted 4 times

 **tfdestroy** 11 months, 3 weeks ago

Selected Answer: B

B. False

Terraform does not restrict who can apply a generated plan based on who created it. As long as the person applying the plan has appropriate permissions to modify the resources in question, they should be able to apply the plan. This means that multiple team members can collaborate on Terraform configurations and one person's plan can be applied by someone else.

However, it's important to note that good practices might involve controlling who has the necessary permissions to apply changes, especially in production environments, to ensure that changes are reviewed and approved before being applied.

upvoted 1 times

 **junk4share** 1 year ago

Selected Answer: B

b is the answer

upvoted 1 times

 **kiran15789** 1 year, 2 months ago

Selected Answer: A

aws_instance.example["].id

is a valid

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

B is the answer

upvoted 1 times

 **Power123** 1 year, 3 months ago

B is correct

upvoted 1 times

 **sahara99** 1 year, 5 months ago

Selected Answer: B

B is correct

upvoted 2 times

 **wimarsha** 1 year, 5 months ago

Someone can run an apply without plan also.

So B is the correct answer.

upvoted 1 times

 **chael88** 1 year, 5 months ago

Selected Answer: B

B. False

Somebody else can apply the plan via Terraform Cloud.

I have tested this.

upvoted 3 times

 **Ahmad_Terraform** 2 years ago

B is correct ,,

upvoted 1 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer.

upvoted 2 times

 **fsdgrtsdfcjmu** 2 years, 1 month ago

It looks like 50% of provided answers here are wrong, should be false

upvoted 2 times

 **biscuithammer** 2 years, 1 month ago

Selected Answer: B

It should be B

upvoted 4 times

 **ItaloVinodi** 2 years, 2 months ago

Question #46

Topic 1

Examine the following Terraform configuration, which uses the data source for an AWS AMI.

What value should you enter for the ami argument in the AWS instance resource?

```
data "aws_ami" "ubuntu" {
  ...
}

resource "aws_instance" "web" {
  ami = _____
  instance_type = "t2.micro"

  tags = {
    Name = "HelloWorld"
  }
}
```

- A. aws_ami.ubuntu
- B. data.aws_ami.ubuntu
- C. data.aws_ami.ubuntu.id
- D. aws_ami.ubuntu.id

Correct Answer: C

```
resource "aws_instance" "web" {
  ami = data.aws_ami.ubuntu.id
}

Reference:  
https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance
```

Community vote distribution

C (100%)

✉  **Jaro3000**  2 years, 1 month ago

C is correct.

upvoted 11 times

✉  **Eltooth**  2 years ago

Selected Answer: C

C is correct answer.

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance>

upvoted 6 times

✉  **Bere**  11 months, 2 weeks ago

Selected Answer: C

In the context of the example given in the question:

```
data "aws_ami" "ubuntu" {
most_recent = true

filter {
name = "name"
values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-*"]
}

filter {
name = "virtualization-type"
values = ["hvm"]
}

owners = ["099720109477"] # Canonical
}

resource "aws_instance" "web" {
ami = data.aws_ami.ubuntu.id
instance_type = "t2.micro"

tags = {
Name = "HelloWorld"
}
}
```

upvoted 4 times

✉  **Ni33** 1 year, 2 months ago

Selected Answer: C

C is the correct answer

upvoted 1 times

Question #47

Topic 1

FILL BLANK -

You need to specify a dependency manually.

What resource meta-parameter can you use to make sure Terraform respects the dependency?

Type your answer in the field provided. The text field is not case-sensitive and all variations of the correct answer are accepted.

Correct Answer: *depends_on*

✉  **habros**  2 years, 1 month ago

depends_on

upvoted 17 times

✉  **Cololand** 2 months ago

depends_on

upvoted 1 times

✉️  **vibzr2023** Most Recent 3 months, 2 weeks ago

depends_on

In Terraform, sometimes the order in which resources are created, updated, or destroyed is important because some resources might rely on others. For example, you might not want a virtual machine to be created before the network it relies on is set up. However, there are situations where Terraform cannot infer dependencies from the configuration alone, especially when the dependency is indirect or based on operational considerations outside of Terraform's knowledge.

The depends_on meta-parameter is used for this purpose. You can add depends_on to any resource block to explicitly specify that the resource depends on other resources. Terraform will ensure that the resources listed in depends_on are created before the resource that defines the dependency.

upvoted 2 times

✉️  **vibzr2023** 3 months, 2 weeks ago

Suppose you have a scenario where you need to ensure a logging service is up and running before your main application starts, but there's no direct reference between the two in your Terraform configuration. You could use depends_on to define this relationship:

```
resource "aws_cloudwatch_log_group" "example" {  
  name = "example-log-group"  
  # Other configuration...  
}
```

```
resource "aws_instance" "app_server" {  
  ami = "ami-123456"  
  instance_type = "t2.micro"  
  # Other configuration...
```

```
  # Manually specify dependency on the CloudWatch Log Group  
  depends_on = [aws_cloudwatch_log_group.example]  
}
```

upvoted 1 times

✉️  **vibzr2023** 3 months, 2 weeks ago

In the above example, even though the aws_instance.app_server doesn't directly reference the aws_cloudwatch_log_group.example, we've used depends_on to tell Terraform that the app_server should only be created after the example-log-group CloudWatch Log Group is created. This ensures that your logging infrastructure is in place before your application starts running.

depends_on is particularly useful in cases where Terraform cannot automatically determine the correct order for creating, updating, or destroying resources based on the configuration alone.

upvoted 1 times

✉️  **jutove_mi** 6 months, 1 week ago

what is the question??

upvoted 3 times

✉️  **Bengi** 6 months, 3 weeks ago

https://developer.hashicorp.com/terraform/language/meta-arguments/depends_on

The depends_on meta-argument instructs Terraform to complete all actions on the dependency object (including Read actions) before performing actions on the object declaring the dependency. When the dependency object is an entire module, depends_on affects the order in which Terraform processes all of the resources and data sources associated with that module.

upvoted 1 times

✉️  **KrishTeam** 10 months ago

did anyone got this question in exam

upvoted 1 times

✉️  **filled** 9 months, 1 week ago

sure did glad to know I at least got this one correct

upvoted 1 times

✉️  **joyboy23** 1 year ago

depends_on, Not depend_on
upvoted 2 times

✉️  **gspb** 1 year, 2 months ago

depends_on
upvoted 1 times

✉️  **Power123** 1 year, 3 months ago

depends_on
upvoted 1 times

 **thor7** 1 year, 3 months ago

depends_on

upvoted 1 times

 **sjoe** 1 year, 3 months ago

depends_on

upvoted 1 times

 **Bilalglg93350** 1 year, 4 months ago

depends_on

upvoted 1 times

 **kennynelcon** 1 year, 6 months ago

depends_on

upvoted 1 times

 **G4Exams** 1 year, 8 months ago

explicit dependencies are set with "depends_on". No doubt.

upvoted 2 times

 **ddewit** 1 year, 10 months ago

depends_on

upvoted 3 times

 **kopper2019** 1 year, 11 months ago

depends_on

upvoted 3 times

 **svsilence** 2 years ago

depends_on

upvoted 4 times

 **Eltooth** 2 years ago

"depends_on" is correct answer.

upvoted 4 times

You have never used Terraform before and would like to test it out using a shared team account for a cloud provider. The shared team account already contains

15 virtual machines (VM). You develop a Terraform configuration containing one VM, perform terraform apply, and see that your VM was created successfully.

What should you do to delete the newly-created VM with Terraform?

- A. The Terraform state file contains all 16 VMs in the team account. Execute terraform destroy and select the newly-created VM.
- B. The Terraform state file only contains the one new VM. Execute terraform destroy.
- C. Delete the Terraform state file and execute Terraform apply.
- D. Delete the VM using the cloud provider console and terraform apply to apply the changes to the Terraform state file.

Correct Answer: B

Community vote distribution

B (88%)

13%

 **habros** Highly Voted 2 years, 1 month ago

B. Only resources mentioned in terraform will be applied and reflected in state. Resources manually created in console and/or not defined in c will not be captured by Terraform

upvoted 24 times

 **wangchung** Highly Voted 2 years, 1 month ago

The question says " You develop a Terraform configuration containing one VM,"... would that not mean that a different state file therefore the 1 vms would not be registered

upvoted 6 times

 **wangchung** 2 years, 1 month ago

B still.

" A. The Terraform state file contains all 16 VMs in the team account. Execute terraform destroy and select the newly-created VM. " ---- wh does that meant "select the newly-created VM" you can't destroy a single resource, to destroy and recreate a new resource you use terraform taint to mark the resource for deletions and recreation

upvoted 4 times

 **Jaro3000** 2 years, 1 month ago

You can destroy a single resource using --target

But still B as we assume we have a separate state file with our vm only

upvoted 2 times

 **Molly1994** Most Recent 1 month, 1 week ago

B or D ? if you delete the one VM in UI, when you apply, it will refresh the state file, the VM has been deleted.

upvoted 1 times

-  **vibzr2023** 3 months, 2 weeks ago
B. The Terraform state file only contains the one new VM. Execute terraform destroy: This is the correct approach. When you create resources with Terraform, it tracks those resources in a state file. Since you've only created one VM with Terraform, only that VM is tracked in the state file. Running terraform destroy will remove all resources tracked in the state file, which in this case, is just the one VM you've created.
upvoted 2 times
-  **vibzr2023** 3 months, 2 weeks ago
A. The Terraform state file contains all 16 VMs in the team account. Execute terraform destroy and select the newly-created VM: This option is incorrect because Terraform's state file does not automatically track resources that were not created or imported through Terraform. Since 15 other VMs in the account were not managed by this Terraform configuration, they will not be in the state file, and terraform destroy will not affect them.
C. Delete the Terraform state file and execute Terraform apply: This approach is not recommended. Deleting the state file removes Terraform's knowledge of the resources it manages, but does not actually delete those resources from the cloud provider. Running terraform apply afterwards would attempt to create the resources anew based on the state file.
upvoted 1 times
-  **vibzr2023** 3 months, 2 weeks ago
D. Delete the VM using the cloud provider console and terraform apply to apply the changes: This option is not advisable because it involves manually deleting a resource outside of Terraform, which can lead to discrepancies between your real infrastructure and Terraform's state. If a resource is manually deleted, Terraform's state file still thinks the resource exists, and running terraform apply will likely result in Terraform trying to "reconcile" the state by re-creating the deleted VM, which is not the intended outcome.
upvoted 1 times
-  **gofavad926** 9 months, 2 weeks ago
Selected Answer: B
B, destroy the created instance
upvoted 1 times
-  **Ni33** 1 year, 2 months ago
Selected Answer: B
B is the correct answer. Terraform only maintains desired state and not actual state of the infrastructure in the account.
upvoted 2 times
-  **Power123** 1 year, 3 months ago
B is the answer
upvoted 1 times
-  **oab720** 1 year, 6 months ago
Selected Answer: B
Don't worry, you won't delete prod instances
upvoted 3 times
-  **FarziWaliMarzi** 1 year, 6 months ago
mostly hypothetically assumed answers, not solid concrete answer here? I somehow feel that its a very poorly framed question.
upvoted 1 times
-  **Network_1** 1 year, 6 months ago
You created a Terraform configuration containing 1 VM... You didn't modify the existing Terraform configuration. Hence, only the config you created would be deleted
upvoted 1 times
-  **pmzone** 1 year, 9 months ago
The single Vm can be deleted using -target option in terraform destroy. In this case, I am assuming that the 15 VM's are created using TF.
upvoted 1 times
-  **therealquan** 1 year, 11 months ago
Selected Answer: B
Definitely B
upvoted 3 times
-  **therealquan** 1 year, 11 months ago
the other existing VMs need to be imported to be affected by the terraform destroy
upvoted 3 times

✉  **yuvifose** 1 year, 12 months ago

Selected Answer: B

Since the other machines are not in Terraform yet, it won't touch them
upvoted 2 times

✉  **Ahmad_Terraform** 2 years ago

because its shared ac
So A is correct
upvoted 1 times

✉  **Eltooth** 2 years ago

Selected Answer: B

B is correct answer.

upvoted 2 times

Question #49

Topic 1

What is the name assigned by Terraform to reference this resource?

```
resource "azurerm_resource_group" "dev" {  
    name = "test"  
    location = "westus"  
}
```

- A. dev
- B. azurerm_resource_group
- C. azurerm
- D. test

Correct Answer: A

Community vote distribution

A (83%)

Other

✉  **tipzzz**  2 years, 2 months ago

Selected Answer: A

dev for sure
upvoted 13 times

✉  **Chuddysambo**  1 year, 1 month ago

the keyword here is "assigned by terraform", the correct answer is B
upvoted 10 times

✉  **vibzr2023** 3 months, 2 weeks ago

"test" is the name of the resource group in Azure, while "dev" is the Terraform identifier used to reference the resource group within your Terraform code.
upvoted 5 times

✉  **mattuyghur**  5 months, 1 week ago

azurerm_resource_group
upvoted 1 times

✉  **hoangphan** 5 months, 1 week ago

Selected Answer: A

A is the correct answer.
- azurerm_resource_group is the resource type
- web is the name of the defined resource
upvoted 2 times

✉️  **Alandt** 5 months, 3 weeks ago

Selected Answer: B

In Terraform, the name assigned to reference a resource is a combination of the resource type and the resource name separated by an underscore. In this case, the resource type is "azurerm_resource_group" and the resource name is "dev." Therefore, the correct answer is:

B. azurerm_resource_group

When referencing this resource in other parts of your Terraform configuration, you would use "azurerm_resource_group.dev" to refer to this specific resource.

upvoted 1 times

✉️  **Alandt** 5 months, 3 weeks ago

Chat-GTP:

Why is A not the answer?

Apologies for the confusion. You are correct, and I appreciate your clarification. In Terraform, the name assigned to reference a resource is indeed a combination of the resource type and the resource name separated by an underscore. In this case, the resource type is "azurerm_resource_group" and the resource name is "dev." Therefore, both A and B could be considered correct:

A. dev
B. azurerm_resource_group

When referencing this resource in other parts of your Terraform configuration, you could use either "dev" or "azurerm_resource_group.dev" refer to this specific resource.

upvoted 1 times

✉️  **gold4otas** 6 months, 2 weeks ago

Selected Answer: B

The keyword here is "assigned by terraform". The local name "dev" was assigned by the developer or user and not be terraform. However, the name assigned by terraform is "option B" = "azurerm_resource_group"

upvoted 1 times

✉️  **Stargazer11** 8 months, 2 weeks ago

Answer A.

A typical resource identifier format in Terraform is: <PROVIDER>_<TYPE>.<NAME>

<PROVIDER> is the name of the provider for the resource (e.g., "aws" for AWS resources).

<TYPE> is the resource type (e.g., "instance" for an AWS EC2 instance).

<NAME> is an optional name assigned to the resource instance (e.g., "my_instance").

upvoted 1 times

✉️  **saotayo** 8 months, 2 weeks ago

Selected Answer: B

"assigned by terraform", not user

upvoted 1 times

✉️  **gofavad926** 9 months, 2 weeks ago

Selected Answer: A

A, dev

upvoted 1 times

✉️  **AWS_cert2023** 1 year, 1 month ago

A is the answer.

<https://developer.hashicorp.com/terraform/language/resources/syntax>

upvoted 1 times

✉️  **Beast_Hollow** 1 year, 1 month ago

Selected Answer: A

Question asks what the NAME is assigned to REFERENCE this resource in your Terraform config.

Resource block consists of a resource type and a reference name hence the answer is A: dev

upvoted 3 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

A is the correct answer.

upvoted 2 times

 **karendavtyan** 1 year, 2 months ago

Selected Answer: D

D. for sure

upvoted 1 times

 **Rezi** 1 year, 3 months ago

B for sure.

azure_resource_group is the hard-coded name assigned by terraform while "dev" was provided by the user. Also, 'dev' could have been any other string value whereas 'azure_resource_group' is a constant. That's how terraform recognizes that resource.

upvoted 4 times

 **Midas_Tepes** 1 year, 3 months ago

The answer is B, The azure_resource_group is the name given by terraform. Dev is a managed name...basically what you are calling this insta

upvoted 1 times

 **Power123** 1 year, 3 months ago

Answer is A - dev

upvoted 1 times

 **Swissmali** 1 year, 3 months ago

Isn't that noticeable that the name assigned by terraform is B. We determine the rest. (dev or test). The key word is assigned by terraform

upvoted 1 times

 **Swissmali** 1 year, 3 months ago

I reviewed some learning material and the correct answer is A.

upvoted 1 times

Question #50

Topic 1

Setting the TF_LOG environment variable to DEBUG causes debug messages to be logged into syslog.

A. True

B. False

Correct Answer: A

Reference:

<https://www.terraform.io/docs/internals/debugging.html>

Debugging Terraform

Hands-on: Try the [Create Dynamic Expressions](#) tutorial on HashiCorp Learn.

Terraform has detailed logs which can be enabled by setting the `TF_LOG` environment variable to any value. This will cause detailed logs to appear on stderr.

You can set `TF_LOG` to one of the log levels `TRACE`, `DEBUG`, `INFO`, `WARN` or `ERROR` to change the verbosity of the logs.

Setting `TF_LOG` to `JSON` outputs logs at the `TRACE` level or higher, and uses a parseable JSON encoding as the formatting.

Community vote distribution

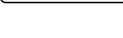
B (82%)

A (18%)

✉️  **Oskar_Madin**  2 years ago

please stop writing or suggestions if you don't know the answer or are not sure!

upvoted 63 times

✉️  **esandips**  2 years, 2 months ago

B

`TF_LOG_PATH` needs to be set as well

upvoted 22 times

✉️  **bigboi23** 2 years, 1 month ago

No.

A is correct.

To persist logged output you can set `TF_LOG_PATH` in order to force the log to always be appended to a specific file when logging is enabled. Note that even when `TF_LOG_PATH` is set, `TF_LOG` must be set in order for any logging to be enabled.

upvoted 2 times

✉️  **scepticemia** 2 years, 2 months ago

Incorrect. A is the correct answer. `TF_LOG_PATH` just enables persistent logging to be appended to a specific file.

From the docs:

To persist logged output you can set `TF_LOG_PATH` in order to force the log to always be appended to a specific file when logging is enabled.

<https://www.terraform.io/internals/debugging>

upvoted 4 times

✉️  **nhatne** 2 years ago

B is correct because

`TF_LOG_PATH` IS NOT REQUIRED, in the docs, they do not mention HAVE TO SET `TF_LOG_PATH`, it is optional, therefore without `TF_LOG_PATH` will cause detailed logs to appear on stderr.

upvoted 5 times

✉️  **starkonbullet**  1 month, 4 weeks ago

Selected Answer: B

B. False.

Setting the `TF_LOG` environment variable to "DEBUG" does not cause debug messages to be logged into syslog. It actually causes the Terraform command-line tool to emit detailed debug output to the console (standard error output, specifically).

upvoted 2 times

 **kingfighters** 3 months, 2 weeks ago

B

Terraform has detailed logs that you can enable by setting the TF_LOG environment variable to any value. Enabling this setting causes detailed logs to appear on stderr.

<https://developer.hashicorp.com/terraform/internals/debugging>

upvoted 1 times

 **vibzr2023** 3 months, 2 weeks ago

B is correct

Setting the TF_LOG environment variable to DEBUG does indeed enable detailed debug messages from Terraform, but these messages are not automatically logged to syslog. By default, when TF_LOG is set, Terraform logs messages to stderr. To direct these logs to a file or another logging destination like syslog, you would need to manually redirect the output or use additional tools or settings specific to your operating system or environment.

For example, in a Unix-like environment, you could redirect the Terraform command's stderr to a file or to syslog using command-line redirect or tools like logger. But this is not something Terraform does automatically just by setting TF_LOG to DEBUG.

To log Terraform's output to a file, you might run a command like:

TF_LOG=DEBUG terraform apply 2>terraform-debug.log

upvoted 2 times

 **Siva_7282** 5 months, 2 weeks ago

Selected Answer: B

B. False

Setting the TF_LOG environment variable to DEBUG does not automatically log debug messages into syslog. Instead, it causes Terraform to print debug messages to the standard error (stderr) output. If you want to capture or redirect these debug messages, you need to handle the stderr output accordingly, such as redirecting it to a file or using other logging mechanisms.

upvoted 7 times

 **SauravAmuze** 5 months, 2 weeks ago

Terraform has detailed logs that you can enable by setting the TF_LOG environment variable to any value. Enabling this setting causes detailed logs to appear on 'stderr'. There is no mention of 'syslog' in the manual. So the answer is B (False)

upvoted 1 times

 **DianaPopal** 6 months, 2 weeks ago

B. False

Setting the TF_LOG environment variable to DEBUG in Terraform does not cause debug messages to be logged into syslog. The TF_LOG environment variable is used to control the logging level of Terraform itself, not the system's syslog service. When TF_LOG is set to DEBUG, Terraform will output detailed debug messages to the standard error output (stderr) or the log file specified in the Terraform configuration.

upvoted 3 times

 **MisterROBOT** 8 months, 2 weeks ago

The answer is False. Setting the TF_LOG environment variable to DEBUG causes debug messages to be logged to the standard error stream (stderr). It does not log messages to syslog.

Syslog is a standard logging system that is used by many Unix-like operating systems to collect and store system log messages. Syslog messages are typically stored in a file called /var/log/syslog. Terraform does not log messages to syslog by default.

To log Terraform messages to syslog, you can use a third-party logging tool, such as Logstash or Fluentd. These tools can be configured to collect logs from stderr and send them to syslog.

Here is an example of how to use Logstash to send Terraform logs to syslog:

```
input {  
  stdin {  
    type => "plain"  
    filter => {  
      remove_field => ["message.level"]  
      add_field => { "message.level" => "debug" }  
    }  
  }  
}  
  
output {  
  syslog {  
    host => "localhost"  
  }  
}
```

upvoted 8 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: A

A, it appears in stderr. As documentation says: "Terraform has detailed logs that you can enable by setting the TF_LOG environment variable to any value. Enabling this setting causes detailed logs to appear on stderr."

upvoted 2 times

 **Halimb** 10 months, 2 weeks ago

Selected Answer: B

<https://developer.hashicorp.com/terraform/cli/config/environment-variables>

Quote "Enables detailed logs to appear on stderr which is useful for debugging."

By default, regardless of the log level, Terraform writes its log output to the standard error (stderr) stream. This behavior is useful for capturing messages in the terminal or console where you are running Terraform commands, allowing you to see any warnings or errors in real-time.

If you wish to redirect the log output to a file or another destination, you can use standard shell redirection as shown in my previous response. This allows you to save the log information for later analysis or to separate it from the terminal output.

upvoted 3 times

 **Spandrop** 10 months, 2 weeks ago

Correct answer is B

"Terraform has detailed logs that you can enable by setting the TF_LOG environment variable to any value. Enabling this setting causes detailed logs to appear on stderr."

There is nothing to do with syslog

<https://developer.hashicorp.com/terraform/internals/debugging>

upvoted 2 times

 **debabratra6983** 10 months, 3 weeks ago

Selected Answer: B

B is the right answer and stop confusing people

upvoted 1 times

 **AWS_cert2023** 1 year, 1 month ago

Enabling this setting causes detailed logs to appear on stderr.

Uses TF_LOG_PATH to change the path of log info.

upvoted 1 times

 **nirosha1** 1 year, 2 months ago

Selected Answer: A

Terraform has detailed logs that you can enable by setting the TF_LOG environment variable to any value. Enabling this setting causes detailed logs to appear on stderr. <https://developer.hashicorp.com/terraform/internals/debugging>

upvoted 2 times

 **AzRNoob** 1 year, 3 months ago

FALSE

Setting the TF_LOG environment variable to DEBUG causes debug messages to be logged to standard error (stderr) output, not to syslog.

upvoted 8 times

 **camps** 1 year, 3 months ago

Selected Answer: B

B. False.

Setting the TF_LOG environment variable to DEBUG causes debug messages to be logged to the console, not to syslog. The TF_LOG environment variable is used to control the level of logging in Terraform. When set to DEBUG, Terraform will log detailed debug messages to the console, which can be useful for troubleshooting issues.

upvoted 4 times

Where in your Terraform configuration do you specify a state backend?

- A. The terraform block
- B. The resource block
- C. The provider block
- D. The datasource block

Correct Answer: A

Backends are configured with a nested backend block within the top-level terraform block.

Reference:

<https://www.terraform.io/docs/language/settings/backends/configuration.html>

Community vote distribution

A (100%)

 **Eltooth** Highly Voted 2 years ago

Selected Answer: A

A is correct answer.

"To configure a backend, add a nested backend block within the top-level terraform block. The following example configures the remote backend."

<https://www.terraform.io/language/settings/backends/configuration>

<https://www.terraform.io/language/settings/backends/configuration#using-a-backend-block>

upvoted 7 times

 **debabrata6983** (Most Recent) 10 months, 3 weeks ago

Selected Answer: A

A is Correct Answer

upvoted 1 times

 **Bere** 11 months, 1 week ago

Selected Answer: A

```
terraform {  
  backend "remote" {  
    hostname = "app.terraform.io"  
    organization = "YourOrganization"
```

```
  workspaces {  
    name = "your_workspace"  
  }  
}
```

upvoted 3 times

 **arunrkaushik** 11 months, 2 weeks ago

```
terraform {  
  backend "consul" {  
    address = "demo.consul.io"  
    scheme = "https"  
    path = "example_app/terraform_state"  
  }  
}
```

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

A for sure.

Question #52

Topic 1

In Terraform 0.13 and above, outside of the required_providers block, Terraform configurations always refer to providers by their local names.

A. True

B. False

Correct Answer: A

Outside of the required_providers block, Terraform configurations always refer to providers by their local names.

Reference:

<https://www.terraform.io/docs/language/providers/requirements.html>

Community vote distribution

A (92%)

8%

 **Bere** Highly Voted 11 months, 1 week ago

Selected Answer: A

Here's an example that demonstrates this:

```

terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
}

provider "aws" {
  region = "us-west-2"
}

resource "aws_instance" "example" {
  ami = "ami-0123456789abcdef0"
  instance_type = "t2.micro"
}

```

In this example, the required_providers block specifies the source and version constraints for the AWS provider. In the rest of the configuration, the provider is referred to by its local name aws, such as in the provider "aws" block and the aws_instance resource block.

upvoted 10 times

 **Eltooth** Highly Voted 2 years ago

Selected Answer: A

A is correct answer.

"Local names are module-specific, and are assigned when requiring a provider. Local names must be unique per-module. Outside of the required_providers block, Terraform configurations always refer to providers by their local names."

<https://www.terraform.io/language/providers/requirements#local-names>

upvoted 8 times

 **BJ5** Most Recent 3 months, 1 week ago

Selected Answer: B

What about aliases

```

provider "aws" {
  region="us-west-1"
  alias = "west"
}

```

```

resource "aws_instance" "ec2" {
  provider = aws.west
}

```

upvoted 1 times

 **chaoscreator** 1 month, 2 weeks ago

that's still referencing the provider name "aws" first

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: A

A. True.

In Terraform 0.13 and above, provider references are always specified using their local name, rather than the legacy name-based syntax used earlier versions. The provider's local name is specified in the required_providers block, and this local name is used in all provider references in configuration.

upvoted 1 times

 **Power123** 1 year, 3 months ago

A is correct

upvoted 1 times

 **AShahine21** 1 year, 6 months ago

Selected Answer: B

what about alias ? I am going to say false

upvoted 1 times

 **Chehh** 1 year, 10 months ago

"Outside of the required_providers block, Terraform configurations always refer to providers by their local names."
<https://www.terraform.io/language/providers/requirements#local-names>

upvoted 3 times

 **bigboi23** 2 years, 1 month ago

Selected Answer: A

Correct!

<https://www.terraform.io/language/providers/requirements>

upvoted 4 times

Question #53

Topic 1

What command should you run to display all workspaces for the current configuration?

- A. terraform workspace
- B. terraform workspace show
- C. terraform workspace list
- D. terraform show workspace

Correct Answer: C

terraform workspace list

The command will list all existing workspaces.

Reference:

<https://www.terraform.io/docs/cli/commands/workspace/list.html>

Community vote distribution

C (100%)

 **bicycle** Highly Voted 2 years, 1 month ago

Selected Answer: C

The terraform workspace list command is used to list all existing workspaces.
upvoted 7 times

 **kcw6** Highly Voted 1 year, 11 months ago

terraform workspace -help
Usage: terraform [global options] workspace

new, list, show, select and delete Terraform workspaces.

Subcommands:

delete Delete a workspace
list List Workspaces
new Create a new workspace
select Select a workspace
show Show the name of the current workspace

upvoted 6 times

Question #54

Topic 1

Terraform providers are always installed from the Internet.

A. True

B. False

Correct Answer: B

Terraform configurations must declare which providers they require, so that Terraform can install and use them.

Reference:

<https://www.terraform.io/docs/language/providers/configuration.html>

Community vote distribution

B (93%)

7%

 **amrith501** Highly Voted 2 years, 1 month ago

Selected Answer: B

For air gapped systems, we can bundle provider along with the Terraform binaries. These Bundle can be installed without internet on air gapped system.

NOTE: For downloading the bundle we need internet access, this can be done on a system which has internet access and later transfer to the air gapped system

upvoted 8 times

 **bigboi23** Highly Voted 2 years, 1 month ago

Terraform CLI finds and installs providers when initializing a working directory. It can automatically download providers from a Terraform registry or load them from a local mirror or cache. If you are using a persistent working directory, you must reinitialize whenever you change a configuration's providers.

upvoted 7 times

 **Tyler2023** Most Recent 8 months, 3 weeks ago

Downloading a plugin directly from its origin registry is not always appropriate, though. For example, the system where you are running Terraform may not be able to access an origin registry due to firewall restrictions within your organization or your locality.

To allow using Terraform providers in these situations, there are some alternative options for making provider plugins available to Terraform which we'll describe in the following sections.

<https://developer.hashicorp.com/terraform/language/providers/requirements#in-house-providers>

<https://developer.hashicorp.com/terraform/cli/config/config-file#provider-installation>

upvoted 2 times

✉  **debabrata6983** 10 months, 3 weeks ago

Selected Answer: B

terraform can be used in AirGap region too
upvoted 1 times

✉  **Bere** 11 months, 1 week ago

Selected Answer: B

Terraform providers are not always installed from the Internet. While Terraform does fetch providers from the HashiCorp Terraform Registry by default, it also allows for the usage of providers from other sources or even local sources.

Here's an example:

```
terraform {  
  required_providers {  
    aws = {  
      source = "hashicorp/aws"  
      version = "2.70.0"  
    }  
    custom-provider = {  
      source = "/path/to/local/custom-provider"  
    }  
  }  
}  
  
provider "aws" {  
  region = "us-west-2"  
}  
  
provider "custom-provider" {  
  # Configuration for the custom provider  
}
```

In this example, the AWS provider is fetched from the HashiCorp Terraform Registry, while the "custom-provider" is loaded from a local path. The local provider must be properly compiled and placed in the specified path.

upvoted 3 times

✉  **LunarPhobia** 11 months, 3 weeks ago

B because you can do air gap installs for networks without internet
upvoted 1 times

✉  **Ni33** 1 year, 2 months ago

Selected Answer: A

A. Terraform Cloud and Terraform Enterprise install providers as part of every run. Terraform CLI finds and installs providers when initializing a working directory.

upvoted 1 times

✉  **vibzr2023** 3 months, 2 weeks ago

incorrect: think of military/defense systems, nuclear power plants they do air-gapped which they don't rely on internet.
upvoted 1 times

✉  **Power123** 1 year, 3 months ago

The answer is B
upvoted 1 times

✉  **vikramv1r** 1 year, 7 months ago

Answer is B.
Terraform CLI finds and installs providers when initializing a working directory. It can automatically download providers from a Terraform registry or load them from a local mirror or cache. If you are using a persistent working directory, you must reinitialize whenever you change a configuration's providers.

upvoted 3 times

✉  **Ahmad_Terraform** 2 years ago

B is correct,
upvoted 1 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer.
upvoted 2 times

Question #55

Topic 1

Which of these is the best practice to protect sensitive values in state files?

- A. Blockchain
- B. Secure Sockets Layer (SSL)
- C. Enhanced remote backends
- D. Signed Terraform providers

Correct Answer: C

Use of remote backends and especially the availability of Terraform Cloud, there are now a variety of backends that will encrypt state at rest and will not store the state in cleartext on machines running.

Reference:

<https://www.terraform.io/docs/extend/best-practices/sensitive-state.html>

Community vote distribution

C (100%)

 **chimons** Highly Voted 🏆 1 year, 6 months ago

Selected Answer: C

The best practice for protecting sensitive values in Terraform state files is to use enhanced remote backends, such as those that support encryption at rest and in transit.

A remote backend allows you to store your Terraform state in a central location, such as an S3 bucket or a HashiCorp Consul server. This allows you to share state files between team members and across multiple environments, such as staging and production.

Using an enhanced remote backend with encryption at rest and in transit helps to protect the sensitive values in your state file from unauthorized access or tampering.

Option C, enhanced remote backends, is the correct answer.

Options A and B, blockchain and SSL, are not directly related to protecting sensitive values in Terraform state files.

Option D, signed Terraform providers, is a security feature that allows you to verify the authenticity of a Terraform provider, but it is not directly related to protecting sensitive values in state files.

upvoted 17 times

Question #56

Topic 1

When does terraform apply reflect changes in the cloud environment?

- A. Immediately
- B. However long it takes the resource provider to fulfill the request
- C. After updating the state file
- D. Based on the value provided to the -refresh command line argument
- E. None of the above

Correct Answer: B

Community vote distribution

B (92%)

8%

 **Nunyabiznes** Highly Voted 🏆 1 year, 3 months ago

Selected Answer: B

When you execute terraform apply, Terraform creates a new execution plan by comparing the current state file to the desired state declared in configuration. After creating the execution plan, Terraform presents the proposed changes and asks for confirmation to apply them. Once you confirm the changes, Terraform updates the state file with the new state reflecting the changes that were made. Terraform then submits the change requests to the resource provider to make the desired changes in the cloud environment. The amount of time it takes for the resource provider to fulfill the requests can vary depending on the resources being modified.

upvoted 14 times

 **liuyomz** 2 months, 3 weeks ago

yeah, B is not exactly correct, but the best one.

upvoted 1 times

 **vitasac** Highly Voted 🏆 2 years, 2 months ago

Selected Answer: B

B for sure

upvoted 11 times

 **foreverlearner** Most Recent 1 year ago

From <https://developer.hashicorp.com/terraform/tutorials/cli/apply>

When you apply this configuration, Terraform will:

- 1) Lock your project's state
- 2) Create a plan, and wait for you to approve it.
- 3) Execute the steps defined in the plan using the providers you installed when you initialized your configuration. Terraform executes steps in parallel when possible, and sequentially when one resource depends on another.
- 4) Update your project's state file with a snapshot of the current state of your resources.
- 5) Unlock the state file.
- 6) Print out a report of the changes it made, as well as any output values defined in your configuration.

Changes in the cloud environment are made on step 3, before updating the state file

upvoted 6 times

 **zanhsieh** 1 year, 2 months ago

Selected Answer: B

I vote for B. The difference between B and C is that we have to count on timeout cases in the real world, e.g. provision GCP DataProc cluster, MongoDB Atlas, etc. It's possible the provisioner timeout but the provider keeps going till the job finish. In this case the local terraform state file does not sync with the physical infra - terraform did try to update but timeout. Then we run 'terraform apply' again and the state file reconciles with the physical infra.

upvoted 2 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

BBBBBBBBBBBB

upvoted 2 times

 **Power123** 1 year, 3 months ago

B is the correct ans

upvoted 1 times

 **Chaks1985** 1 year, 6 months ago

Ans: C ("WHEN" is key in this question. Its not asking how long will it take but "WHEN")

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

Selected Answer: B

If you are creating a new virtual machine using Terraform, it may take a few minutes for the virtual machine to be created and for it to become available for use. During this time, Terraform will continue to report on the progress of the creation process and will display any errors or issues that may arise. Once the virtual machine has been successfully created, the changes will be reflected in your cloud environment. I go with B

upvoted 2 times

 **FarziWaliMarzi** 1 year, 6 months ago

Given it has more than 4 options, I think it should have multiple valid answers, and hence both B & C should be selected. Again, this is assumption based as this is missing in question.

upvoted 1 times

 **faltu1985** 1 year, 9 months ago

Selected Answer: B

B for sure

upvoted 1 times

 **Nightducky** 1 year, 9 months ago

Why B? You have "when" not "how long" in the question.

upvoted 1 times

 **RVivek** 1 year, 9 months ago

Selected Answer: C

When does terraform apply reflect changes in the cloud environment?

Terraform will apply changes to the cloud, then it will update the state file. After it updates the statefile only it "reflects" those changes to the screen

upvoted 1 times

 **pas77** 1 year, 8 months ago

This is not what the question is asking. The question is not asking when it will reflect the changes in the "screen". It is asking when it will reflect the changes in the "cloud environment". And this will be whenever it finishes deploying the resources. "B" is the correct answer for sure.

upvoted 2 times

 **nhatne** 2 years ago

Selected Answer: C

I vote C because the question is "When" and from the docs:

Terraform plan and apply operations run an implicit in-memory refresh as part of their functionality, reconciling any drift from your state file before suggesting infrastructure changes.

upvoted 2 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer.

upvoted 1 times

 **Zam88** 2 years ago

B is correct

upvoted 1 times

 **AzureGurl** 2 years, 1 month ago

~~B is the correct answer~~

Question #57

Topic 1

How would you reference the "name" value of the second instance of this fictitious resource?

```
resource "aws_instance" "web" {
  count = 2
  name = "terraform-${count.index}"
}
```

- A. element(aws_instance.web, 2)
- B. aws_instance.web[1].name
- C. aws_instance.web[1]
- D. aws_instance.web[2].name
- E. aws_instance.web.*.name

Correct Answer: A

Reference:

<https://www.terraform.io/docs/configuration-0-11/interpolation.html>

Community vote distribution

B (97%)

2%

 **franksoul** Highly Voted 2 years, 1 month ago

Dear ALL, tested in Lab and Answer is : B

upvoted 26 times

 **bigboi23** Highly Voted 2 years, 1 month ago

Selected Answer: B

B!!!!!!

upvoted 15 times

 **nhaastrup** Most Recent 2 months, 2 weeks ago

To reference the "name" value of the second instance of the fictitious resource "aws_instance" named "server", you can use the element index notation in Terraform. The element index notation allows you to access elements in a list or collection by their index.

Given that the index in Terraform starts from 0, the second instance has an index of 1. Therefore, to reference the "name" value of the second instance = B

upvoted 3 times

 **kingfighters** 3 months, 2 weeks ago

A. element(aws_instance.web, 2)

is correct if we want to access the 3rd one, but there is no "the third one"

upvoted 2 times

 **samimshaikh** 6 months, 2 weeks ago

Selected Answer: B

because the index starts from 0 so second instance name will be at 1

upvoted 4 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: B

B for sure

upvoted 1 times

 **debabrata6983** 10 months, 3 weeks ago

Selected Answer: B

This is the very basics of terraform resource referencing

upvoted 3 times

 **Bere** 11 months, 1 week ago

Selected Answer: B

Example:

```
resource "aws_instance" "web" {  
  count = 2  
  name = "terraform-${count.index}"  
}
```

```
output "second_instance_name" {  
  value = aws_instance.web[1].name  
}
```

upvoted 8 times

 **Bluemoon22** 1 year, 2 months ago

The answer is B, aws_instance.web[1].name

upvoted 5 times

 **Power123** 1 year, 3 months ago

B is correct

upvoted 2 times

 **thor7** 1 year, 3 months ago

Selected Answer: B

B is a correct answer, checked.

upvoted 1 times

 **Nunyabiznes** 1 year, 3 months ago

Selected Answer: B

The count.index starts from 0. Therefore, the second instance would have an index of 1. In that case, the correct answer would be B. aws_instance.web[1].name.

upvoted 9 times

 **Ell89** 1 year, 4 months ago

Selected Answer: B

its B - people who answer incorrectly with such conviction and belief are dangerous.

upvoted 4 times

 **oab720** 1 year, 6 months ago

Selected Answer: B

Tried and tested, definitely B

upvoted 1 times

 **Bebins** 1 year, 6 months ago

The output of A won't return the name of the 2nd instance. So here the answer is "B"

upvoted 1 times

 **adouban** 1 year, 7 months ago

Selected Answer: B

B is the correct answer

upvoted 1 times

 **asudhin** 1 year, 9 months ago

Selected Answer: B

It is B

upvoted 1 times

Question #58

Topic 1

A Terraform provider is not responsible for:

- A. Understanding API interactions with some service
- B. Provisioning infrastructure in multiple clouds
- C. Exposing resources and data sources based on an API
- D. Managing actions to take based on resource differences

Correct Answer: D

Reference:

<https://www.terraform.io/docs/configuration-0-11/providers.html>

Providers are responsible in Terraform for managing the lifecycle of a **resource**: create, read, update, delete.

Most providers require some sort of configuration to provide authentication information, endpoint URLs, etc. Where explicit configuration is required, a `provider` block is used within the configuration as illustrated in the following sections.

By default, resources are matched with provider configurations by matching the start of the resource name. For example, a resource of type `vsphere_virtual_machine` is associated with a provider called `vsphere`.

Community vote distribution

D (45%)

B (44%)

9%

✉️  **amrith501** Highly Voted 2 years, 1 month ago

Selected Answer: B

The answer should be B

A terraform can only provision resource in one Cloud not multiple cloud

upvoted 15 times

✉️  **Nunyabiznes** 1 year, 3 months ago

No, it can provision in multiple clouds:

```
provider "aws" {  
  region = "us-west-2"  
}
```

```
provider "azurerm" {  
  features {}  
}
```

```
resource "aws_instance" "example" {  
  ami = "ami-0c55b159cbfafe1f0"  
  instance_type = "t2.micro"  
}
```

```
resource "azurerm_resource_group" "example" {  
  name = "example"  
  location = "East US"  
}
```

upvoted 7 times

✉️  **jerikoo** 1 year, 2 months ago

those are 2 providers, each provider can only deploy resources in one cloud.. several providers, in multicloud.. so "A Terraform provider can only provision in one Cloud... B!

upvoted 20 times

✉️  **Zam88** Highly Voted 2 years ago

answer is B

upvoted 10 times

✉️  **Molly1994** Most Recent 1 month, 1 week ago

the answer is definitely D

upvoted 1 times

✉️  **chaoscreator** 1 month, 2 weeks ago

Selected Answer: D

Agree with D

upvoted 2 times

✉️  **vibzr2023** 3 months, 2 weeks ago

D is correct.

I don't know so many votes for B which is pretty basic that terraform can handle multiple providers not providers themselves

upvoted 2 times

✉️  **090200f** 4 days, 20 hours ago

yes me too think in same way.. I vote for D

upvoted 1 times

✉️  **Alandt** 5 months, 2 weeks ago

Selected Answer: D

D: the management of actions based on resource differences is handled by Terraform itself, not the provider. The provider simply informs Terraform about the current state of the resource and how to create, update, or delete it.

upvoted 5 times

✉️  **samimshaikh** 6 months, 2 weeks ago

Selected Answer: D

D.

Managing actions to take based on resource differences: This is not a primary responsibility of a Terraform provider. Instead, Terraform itself handles this by comparing the desired state (defined in Terraform configurations) with the current state of the infrastructure and determining the necessary actions to achieve the desired state

upvoted 5 times

 **TigerInTheCloud** 6 months, 4 weeks ago

Selected Answer: D

B: you can write a provider to work on multiple clouds.
D: is terraform core's work.

upvoted 4 times

 **Spandrop** 7 months ago

Selected Answer: D

I'm not a native English speaker, but I think that in the context of the question, "A Terraform provider" is referring to the concept of providers in Terraform as a whole, not a specific individual provider.

I would go with D

"Terraform providers are a plugin for Terraform that makes a collection of related resources available. A provider plugin is responsible for understanding API interactions and exposing resources. Providers generally are IaaS (like AWS, GCP, Microsoft Azure, OpenStack), PaaS (like Heroku), or SaaS services (like Terraform Cloud, DNSimple, CloudFlare)."

However, the management of actions based on resource differences (i.e., what to create, update, or delete) is handled by Terraform's core engine, not by the providers. The providers simply inform Terraform's core about what resources they can manage and how to manage them.
upvoted 8 times

 **ghostGuiggs** 8 months, 2 weeks ago

Selected Answer: B

B is the answer
upvoted 1 times

 **ealpuche** 9 months ago

D. Managing actions to take based on resource differences.

Terraform providers primarily focus on understanding API interactions with specific services, provisioning infrastructure, exposing resources a data sources based on an API, and maintaining the state of resources. However, managing actions based on resource differences is typically handled by Terraform's core functionality, specifically the `terraform plan` and `terraform apply` commands, which determine the changes neede achieve the desired state and then apply those changes based on the execution plan. Providers interact with the target services but do not manage the core Terraform plan and apply process.

upvoted 3 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: B

GPT answer: B. Provisioning infrastructure in multiple clouds

Terraform providers are responsible for understanding API interactions with some service, exposing resources and data sources based on an and managing actions to take based on resource differences. However, Terraform providers are not responsible for provisioning infrastructure multiple clouds. This is because Terraform is a cloud-agnostic tool, and each cloud provider has its own unique set of APIs and resources.

upvoted 2 times

 **aanataliya** 10 months, 1 week ago

Selected Answer: B

To some people confusion is between B and D. Let me try to clarify.

1. One who rejects B because of one specific provider "megaport". Technically, multi cloud deployment feature is provided by megaport and i by a provider. Quesition is not talking about specific provider. A provider(single provider) cannot mean "megaport" provider. it supposed to be true for any other single provider. so B cannot be rejected.

Ref: https://registry.terraform.io/providers/megaport/megaport/latest/docs/guides/example_multicloud_aws_azure

2. D cannot be answer. As per terraform, providers are plugins. Terraform make a plan for desired state and communicate with plugin(provider) make change. so it is responsibility of provider.

Ref: <https://developer.hashicorp.com/terraform/plugin/how-terraform-works>

upvoted 2 times

 **BaburTurk** 10 months, 3 weeks ago

Selected Answer: B

A Terraform provider is responsible for interacting with a specific API or service to manage resources and infrastructure. It abstracts the API interactions and exposes resources and data sources to be managed through Terraform configurations. It also handles managing actions bas on resource differences (such as creating, updating, or deleting resources). However, it is not responsible for provisioning infrastructure in multiple clouds. This is usually handled by different providers, each tailored to a specific cloud or service.

upvoted 2 times

 **modarov** 11 months, 2 weeks ago

B. Provisioning infrastructure in multiple clouds.

upvoted 2 times

 **LunarPhobia** 11 months, 3 weeks ago

B because a Terraform provider is for single cloud. You'll need several providers to have multiple clouds. Also a Terraform provider is much more than just for cloud use

upvoted 1 times

 **saskuachmukaz** 1 year ago

Selected Answer: D

A Terraform provider is responsible for understanding API interactions with a specific service, provisioning infrastructure in a specific cloud, and exposing resources and data sources based on an API. However, managing actions to take based on resource differences is the responsibility of Terraform's core engine, not the provider itself. The core engine compares the desired state declared in the Terraform configuration with the current state stored in the Terraform state file and determines the actions needed to achieve the desired state, such as creating, updating, or deleting resources.

So, the answer is D. Managing actions to take based on resource differences.

upvoted 3 times

Question #59

Topic 1

Terraform provisioners can be added to any resource block.

A. True

B. False

Correct Answer: A

Reference:

<https://www.terraform.io/docs/language/resources/provisioners/syntax.html>

Terraform includes the concept of provisioners as a measure of pragmatism, knowing that there will always be certain behaviors that can't be directly represented in Terraform's declarative model.

However, they also add a considerable amount of complexity and uncertainty to Terraform usage. Firstly, Terraform cannot model the actions of provisioners as part of a plan because they can in principle take any action. Secondly, successful use of provisioners requires coordinating many more details than Terraform usage usually requires: direct network access to your servers, issuing Terraform credentials to log in, making sure that all of the necessary external software is installed, etc.

The following sections describe some situations which can be solved with provisioners in principle, but where better solutions are also available. We do not recommend using provisioners for any of the use-cases described in the following sections.

Even if your specific use-case is not described in the following sections, we still recommend attempting to solve it using other techniques first, and use provisioners only if there is no other option.

Community vote distribution

A (87%)

13%

✉️  **GopinathM** Highly Voted 1 year, 10 months ago

A is correct

<https://www.terraform.io/language/resources/provisioners/syntax>
resource "aws_instance" "web" {
...

```
provisioner "local-exec" {  
command = "echo The server's IP address is ${self.private_ip}"  
}  
}  
upvoted 14 times
```

✉️  **agmesas** Highly Voted 1 year, 5 months ago

Selected Answer: A

the key of this question is "could you use a local-exec provisioner in any type of resource?", yes, because the execution of this provisioner is my local so we could use local-exec in any resource type because it is independent of the type of resource. Answer is A "true"
upvoted 10 times

✉️  **samimshaikh** Most Recent 6 months, 2 weeks ago

Selected Answer: A

A. True

Terraform provisioners can be added to any resource block. Provisioners are used to execute scripts or commands on a local or remote machine as part of resource creation or destruction. They are defined within a resource block and allow you to perform actions such as initializing, configuring, or setting up resources after they have been created.

upvoted 2 times

✉  **Pradh** 9 months, 3 weeks ago

answer should be "B"

Can you add a provisioner to below resource block ?

```
=====
resource "azurerm_network_interface" "Nic-1"
=====
```

upvoted 3 times

✉  **ilm70** 4 months, 3 weeks ago

```
resource "azurerm_network_interface" "Nic-1" {
# ...

provisioner "local-exec" {
  command = "echo 'Network interface created'"
}
}

upvoted 2 times
```

✉  **debabrata6983** 10 months, 3 weeks ago

Selected Answer: A

Please stop confusing people. Basics of Provisioner block : always bind with resource block

upvoted 6 times

✉  **Bere** 11 months, 1 week ago

Selected Answer: A

Since the local-exec provisioner executes on the machine running Terraform and not on the resource itself, it can indeed be added to any resource block. The actions taken by the local-exec provisioner are not dependent on the type of resource, so it's possible to use this provisioner with any resource in Terraform.

upvoted 4 times

✉  **modarov** 11 months, 2 weeks ago

The answer is False. Terraform provisioners can only be added to resource blocks that support them

upvoted 2 times

✉  **LunarPhobia** 11 months, 3 weeks ago

The new exam doesn't ask about provisioners anymore

upvoted 3 times

✉  **Ha_Baruh_Architect13** 1 year ago

The correct answer is B. False.

Terraform provisioners are not added to resource blocks directly. Instead, provisioners are added as separate blocks within a resource block. Provisioners are used to execute scripts or commands on a resource after it has been created or updated. They are defined outside of the resource block and are associated with the resource using the "provisioner" keyword followed by the specific provisioner type CHATGPT

upvoted 1 times

✉  **Jlee7** 1 year ago

The answer is False.

Terraform provisioners can only be added to resource blocks that support them. For example, the AWS provider supports the remote-exec provisioner, but the Azure provider does not.

upvoted 1 times

✉  **joyboy23** 1 year ago

But what about the local-exec ?

upvoted 1 times

✉  **Ni33** 1 year, 2 months ago

Selected Answer: A

AAAAAAA

upvoted 2 times

✉  **Bluemoon22** 1 year, 2 months ago

The answer is A

upvoted 2 times

 **sylvergorilla** 1 year, 3 months ago

Selected Answer: A

Yes, Terraform provisioners can be added to any resource block. Provisioners are used to execute scripts or commands on a resource after it has been created or updated.

upvoted 3 times

 **Power123** 1 year, 3 months ago

A is correct

upvoted 1 times

 **Nunyabiznes** 1 year, 3 months ago

Selected Answer: A

Provisioners can be added to any resource block in Terraform configuration, but it is not recommended to use them for configuration management as it goes against the declarative approach of Terraform. Provisioners should only be used as a last resort when no other Terraform resource types are available to handle a specific task.

upvoted 3 times

 **Tanacet** 1 year, 5 months ago

Selected Answer: A

Provisioner is an arbitrary command executed by the terraform when a resource is created/destroyed. It is not related to a resource itself. The creation/destruction of a resource is just a trigger.

upvoted 3 times

 **FarziWaliMarzi** 1 year, 6 months ago

Selected Answer: A

Option A is common sense

upvoted 1 times

Question #60

Topic 1

What is terraform refresh intended to detect?

- A. Terraform configuration code changes
- B. Empty state files
- C. State file drift
- D. Corrupt state files

Correct Answer: C

Reference:

<https://www.hashicorp.com/blog/detecting-and-managing-drift-with-terraform>

Prior to a plan or apply operation, Terraform does a refresh to update the state file with real-world status. You can also do a refresh any time with `terraform refresh`:

```
$ terraform refresh
aws_instance.example: Refreshing state... (ID: i-011a9893eff09ede1)
```

What Terraform is doing here is reconciling the resources tracked by the state file with the real world. It does this by querying your infrastructure providers to find out what's actually running and the current configuration, and updating the state file with this new information. Terraform is designed to co-exist with other tools as well as manually provisioned resources and so it only refreshes resources under its management.

The output for a refresh is minimal. Terraform lists each resource it is refreshing along with its internal ID. Running `refresh` does not modify infrastructure, but does modify the state file. If the state has drifted from the last time Terraform ran, `refresh` allows that drift to be detected.

Community vote distribution

C (100%)

 **camps**  1 year, 3 months ago

Selected Answer: C

C. State file drift.

The terraform refresh command is used to reconcile the state Terraform has stored in the state file with the real-world infrastructure. When you run terraform apply, Terraform updates the state file to reflect the current state of the infrastructure it manages. However, if changes are made to the infrastructure outside of Terraform, such as via the web console or API, the state file will become out-of-date and will not accurately reflect the current state of the infrastructure.

upvoted 6 times

 **LunarPhobia**  11 months, 3 weeks ago

Question #61

Topic 1

FILL BLANK -

Which flag would you add to terraform plan to save the execution plan to a file?

Type your answer in the field provided. The text field is not case-sensitive and all variations of the correct answer are accepted.

Correct Answer: `-out=FILENAME`

Reference:

<https://www.terraform.io/docs/cli/commands/plan.html>

You can use the optional `-out=FILE` option to save the generated plan to a file on disk, which you can later execute by passing the file to `terraform apply` as an extra argument. This two-step workflow is primarily intended for when running Terraform in automation.

If you run `terraform plan` without the `-out=FILE` option then it will create a *speculative plan*, which is a description of the effect of the plan but without any intent to actually apply it.

 **Eltooth**  2 years ago
Answer is correct: terraform plan -out=FILE

"You can use the optional -out=FILE option to save the generated plan to a file on disk, which you can later execute by passing the file to terraform apply as an extra argument. This two-step workflow is primarily intended for when running Terraform in automation.

If you run terraform plan without the -out=FILE option then it will create a speculative plan, which is a description of the effect of the plan but without any intent to actually apply it."

<https://www.terraform.io/cli/commands/plan>
upvoted 8 times

 **secdaddy**  1 year, 7 months ago
Question asks for the flag and the flag is -out
=FILE is the parameter for the flag
No idea if they'll just parse this for -out or if =FILE will be useful
terrafrom plan -out
| Error: Failed to parse command-line flags
| flag needs an argument: -out
upvoted 5 times

 **Canarys_Automations**  6 months, 1 week ago
-out=FILEPATH
upvoted 1 times

Question #62

Topic 1

FILL BLANK -

What is the name of the default file where Terraform stores the state?

Type your answer in the field provided. The text field is not case-sensitive and all variations of the correct answer are accepted.

Correct Answer: *Terraform.tfstate*

Reference:

<https://www.terraform.io/docs/language/state/index.html>

State

JUMP TO SECTION ▾

Terraform must store state about your managed infrastructure and configuration. This state is used by Terraform to map real world resources to your configuration, keep track of metadata, and to improve performance for large infrastructures.

This state is stored by default in a local file named "terraform.tfstate", but it can also be stored remotely, which works better in a team environment.

 **Eltooth** Highly Voted 2 years ago

Answer is correct : terraform.tfstate

"This state is stored by default in a local file named "terraform.tfstate", but it can also be stored remotely, which works better in a team environment."

<https://www.terraform.io/language/state>

upvoted 7 times

 **090200f** Most Recent 3 days, 19 hours ago

terraform.tfstate , which is stored in our local workspace

upvoted 1 times

 **NPE_** 7 months, 2 weeks ago

The answer is wrong. terraform.tfstate not Terraform.tfstate, with t not T.

upvoted 1 times

 **joyboy23** 1 year ago

terraform.tfstate

upvoted 1 times

 **Power123** 1 year, 3 months ago

terraform.tfstate

upvoted 1 times

 **thor7** 1 year, 3 months ago

terraform.tfstate

upvoted 1 times

 **BilalIg93350** 1 year, 4 months ago

terraform.tfstate

upvoted 2 times

 **titoTitoApo** 1 year, 5 months ago

terraform.tfstate

upvoted 2 times

 **kopper2019** 1 year, 11 months ago

terraform.tfstate

upvoted 4 times

A Terraform local value can reference other Terraform local values.

- A. True
- B. False

Correct Answer: A

Reference:

<https://www.terraform.io/docs/configuration-0-11/locals.html>

The `locals` block defines one or more local variables within a module. Each `locals` block can have as many locals as needed, and there can be any number of `locals` blocks within a module.

The names given for the items in the `locals` block must be unique throughout a module.

The given value can be any expression that is valid within the current module.

The expression of a local value can refer to other locals, but as usual reference cycles are not allowed. That is, a local cannot refer to itself or to a variable that refers (directly or indirectly) back to it.

It's recommended to group together logically-related local values into a single block, particularly if they depend on each other. This will help the reader understand the relationships between variables. Conversely, prefer to define *unrelated* local values in *separate* blocks, and consider annotating each block with a comment describing any context common to all of the enclosed locals.

Community vote distribution

A (100%)

 **Eltooth** Highly Voted 2 years ago

Answer is correct : True

"The expressions in local values are not limited to literal constants; they can also reference other values in the module in order to transform or combine them, including variables, resource attributes, or other local values."

<https://www.terraform.io/language/values/locals#declaring-a-local-value>

upvoted 13 times

 **Molly1994** Most Recent 1 month, 1 week ago

A
locals {
IDs for multiple sets of EC2 instances, merged together
instance_ids = concat(aws_instance.blue.*.id, aws_instance.green.*.id)
}
locals {
Common tags to be assigned to all resources
common_tags = {
Service = local.service_name
Owner = local.owner
}
}

upvoted 2 times

 **ilmi70** 4 months, 3 weeks ago

locals {
common_tags = {
Owner = "team-name"
Service = "customer-service"
}
extra_tags = merge(local.common_tags, { Environment = "production" })
}

upvoted 1 times

 **debabrata6983** 10 months, 3 weeks ago

Selected Answer: A

The right answer is A as local value can also reference other terraform local value(s)

upvoted 1 times

 **Ragque** 10 months, 4 weeks ago

Selected Answer: A

A is the correct answer

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: A

A. True.

Question #64

Topic 1

Which of the following is not a valid Terraform collection type?

- A. list
- B. map
- C. tree
- D. set

Correct Answer: C

Reference:

<https://www.terraform.io/docs/language/expressions/type-constraints.html>

The three kinds of collection type in the Terraform language are:

- `list(...)` : a sequence of values identified by consecutive whole numbers starting with zero.

The keyword `list` is a shorthand for `list(any)`, which accepts any element type as long as every element is the same type. This is for compatibility with older configurations; for new code, we recommend using the full form.

- `map(...)` : a collection of values where each is identified by a string label.

The keyword `map` is a shorthand for `map(any)`, which accepts any element type as long as every element is the same type. This is for compatibility with older configurations; for new code, we recommend using the full form.

Maps can be made with braces (`{}`) and colons (`:`) or equals signs (`=`): `{ "foo": "bar", "bar": "baz" }` OR `{ foo = "bar", bar = "baz" }`. Quotes may be omitted on keys, unless the key starts with a number, in which case quotes are required. Commas are required between key/value pairs for single line maps. A newline between key/value pairs is sufficient in multi-line maps.

Note: although colons are valid delimiters between keys and values, they are currently ignored by `terraform fmt` (whereas `terraform fmt` will attempt vertically align equals signs).

- `set(...)` : a collection of unique values that do not have any secondary identifiers or ordering.

Community vote distribution

C (100%)

 **godie44** 2 months, 3 weeks ago

Selected Answer: C

C is the right answer
upvoted 2 times

 **debabratash983** 10 months, 3 weeks ago

Selected Answer: C

the tree is not a valid collection type in terrafrom
upvoted 1 times

 **SairamObili** 11 months ago

we dont have tree type in terraform
upvoted 1 times

 **Bluemoon22** 1 year, 2 months ago

answer is C, tree
upvoted 2 times

 **Power123** 1 year, 3 months ago

Answer is C - tree
upvoted 1 times

 **Ignotus** 2 years ago

my bad, is NOT a valid... answer C is correct
upvoted 1 times

 **Ignotus** 2 years ago

tree isn't even on that page..
upvoted 2 times

 **Ahmad_Terraform** 2 years ago

C is the right answer
upvoted 4 times

 **Eltooth** 2 years ago

Question #65

Topic 1

When running the command terraform taint against a managed resource you want to force recreation upon, Terraform will immediately destroy and recreate the resource.

A. True

B. False

Correct Answer: B

Reference:

<https://www.devopsschool.com/blog/terraform-taint-and-untaint-explained-with-example-programs-and-tutorials/>

Community vote distribution

B (100%)

 **HereToReAssureMyself** Highly Voted 2 years ago

Selected Answer: B

its not immediately

The terraform taint command informs Terraform that a particular object has become degraded or damaged. Terraform represents this by marking the object as "tainted" in the Terraform state, and Terraform will propose to replace it in the next plan you create.

upvoted 19 times

 **Eltooth** Highly Voted 2 years ago

Selected Answer: B

B is correct answer. False.

"The terraform taint command informs Terraform that a particular object has become degraded or damaged. Terraform represents this by marking the object as "tainted" in the Terraform state, and Terraform will propose to replace it in the next plan you create."

FYI - This command is deprecated. For Terraform v0.15.2 and later, we recommend using the -replace option with terraform apply instead. For Terraform v0.15.2 and later, we recommend using the -replace option with terraform apply to force Terraform to replace an object even though there are no configuration changes that would require it.

<https://www.terraform.io/cli/commands/taint>

upvoted 15 times

 **Molly1994** Most Recent 1 month, 1 week ago

false. taint is marking status, terraform apply make it destroy and recreate

upvoted 2 times

 **ilmi70** 4 months, 3 weeks ago

B. False

The terraform taint command only marks a Terraform-managed resource as tainted, forcing it to be destroyed and recreated on the next apply. However, it does not immediately destroy and recreate the resource. The actual destruction and recreation happen when you run terraform apply

upvoted 4 times

 **debabrata6983** 10 months, 3 weeks ago

Selected Answer: B

It doesn't apply immediately. It will replace the managed resource, which is marked as tainted in the state file, during the next execution cycle (aka terraform apply)

upvoted 2 times

 **BennaniHaythem** 1 year, 3 months ago

True :

When you use the terraform taint command, it marks a resource as "tainted", indicating that it needs to be recreated on the next terraform apply run. Once a resource is marked as tainted, Terraform will destroy and recreate it as part of the next apply operation. This behavior is intended to ensure that the resource is recreated from scratch and that any changes to its configuration are applied properly. Therefore, the statement "When running the command terraform taint against a managed resource you want to force recreation upon, Terraform will immediately destroy and recreate the resource" is true.

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: B

B. False.

When running the command terraform taint against a managed resource in Terraform, you are marking the resource as tainted, which means it will be recreated during the next terraform apply run. Tainting a resource does not immediately destroy and recreate the resource, but it causes Terraform to consider the resource as out-of-date and recreate it when the configuration is next applied.

upvoted 2 times

 **oab720** 1 year, 6 months ago

Selected Answer: B

Can only be changed after 'Terraform apply'

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

next apply.

upvoted 4 times

👤 fifi1907 2 years, 1 month ago

Not A?

The terraform taint command manually marks a Terraform-managed resource as tainted, forcing it to be destroyed and recreated on the next apply.

upvoted 2 times

👤 bp339 2 years, 1 month ago

Terraform will immediately destroy and recreate the resource.. tf doesn't immediately destroy it.. it will just mark it as tainted and in the next plan/apply it will recreate..so answer is B

upvoted 8 times

Question #66

Topic 1

All standard backend types support state storage, locking, and remote operations like plan, apply and destroy.

A. True

B. False

Correct Answer: A

Reference:

<https://www.terraform.io/docs/language/settings/backends/remote.html>

Community vote distribution

B (91%)

9%

👤 vadeemkaa Highly Voted 1 year, 6 months ago

B. False

Backend types:

Enhanced Backend – Additional operations like plan, apply, etc. on remote.

Standard backend – Simple State file storage and lock facility

upvoted 12 times

👤 BaburTurk Highly Voted 10 months, 3 weeks ago

Selected Answer: B

The answer is False.

Not all standard backend types support state locking and remote operations. The following standard backend types do not support state lock

AzureRM

AzureKeyVault

Consul

Docker

Google Cloud Storage

Kubernetes

MySQL

Oracle

PostgreSQL

Vault

The following standard backend types support state locking but not remote operations:

AWS S3

upvoted 7 times

👤 krusty93 9 months ago

Don't know the others, but AzureRM supports state locking: <https://developer.hashicorp.com/terraform/language/settings/backends/azure>

upvoted 2 times

✉  **kingfighters** Most Recent 3 months, 2 weeks ago

Note: In Terraform versions before 1.1.0, we classified backends as standard or enhanced. The enhanced label differentiated the remote backend, which could both store state and perform Terraform operations. This classification has been removed.
<https://developer.hashicorp.com/terraform/language/settings/backends/configuration> another deprecated feature and question..
upvoted 1 times

✉  **NashP** 5 months, 2 weeks ago

A. True

Explanation:

All standard backend types in Terraform support state storage, locking, and remote operations such as plan, apply, and destroy. Terraform backends are responsible for storing the Terraform state file, managing concurrent access to the state, and facilitating collaboration among team members. Different backend types may have additional features or considerations, but these fundamental capabilities are common across all standard backends.

upvoted 1 times

✉  **itstammy** 6 months ago

<https://digitalvarys.com/complete-terraform-tutorial-part-6-terraform-backends/> its B

upvoted 1 times

✉  **MisterROBOT** 8 months, 2 weeks ago

Terraform

Documentation - Not all backends support locking. The documentation for each backend includes details about whether it supports locking or not.

upvoted 2 times

✉  **foreverlearner** 1 year ago

This question is outdated <https://developer.hashicorp.com/terraform/language/settings/backends/configuration> : In Terraform versions before 1.1.0, we classified backends as standard or enhanced. The enhanced label differentiated the remote backend, which could both store state and perform Terraform operations. This classification has been removed.

"Some of these backends act like plain remote disks for state files, while others support locking the state while operations are being performed."

upvoted 2 times

✉  **Ni33** 1 year, 2 months ago

Selected Answer: A

AAAAAAAAAAAAAA

upvoted 2 times

✉  **jerikoo** 1 year, 2 months ago

yes, it's A.

SOME of these backends act like plain remote disks for state files, while OTHERS support locking the state while operations are being performed.

SO not ALL are LOCKING... AI

upvoted 1 times

✉  **jerikoo** 1 year, 2 months ago

sorry, dyslexia, its B! :))

upvoted 2 times

✉  **Power123** 1 year, 3 months ago

Answer is B

upvoted 2 times

✉  **agmesas** 1 year, 5 months ago

B. Artefactory is a standard backend and has not locking feature. Also, etcd has not locking feature (only etcdv3).

upvoted 4 times

✉  **kopper2019** 1 year, 11 months ago

Selected Answer: B

not all of them so False

upvoted 4 times

👤 **Ggggg23** 1 year, 12 months ago

<https://www.terraform.io/language/settings/backends/configuration>

"Some of these backends act like plain remote disks for state files, while others support locking the state while operations are being performed. This helps prevent conflicts and inconsistencies. The built-in backends listed are the only backends. You cannot load additional backends as plugins."

upvoted 3 times

👤 **Ahmad_Terraform** 2 years ago

Not all backend encrypt , B false = correct

upvoted 1 times

👤 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer : False.

"By default, Terraform uses a backend called local, which stores state as a local file on disk. You can also configure one of the built-in backends listed in the documentation sidebar.

Some of these backends act like plain remote disks for state files, while others support locking the state while operations are being performed. This helps prevent conflicts and inconsistencies. The built-in backends listed are the only backends. You cannot load additional backends as plugins."

<https://www.terraform.io/language/settings/backends/configuration#available-backends>

upvoted 4 times

👤 **HDDH** 2 years ago

Selected Answer: B

What Backends Do

Backends primarily determine where Terraform stores its state. Terraform uses this persisted state data to keep track of the resources it manages. Since it needs the state in order to know which real-world infrastructure objects correspond to the resources in a configuration, everyone working with a given collection of infrastructure resources must be able to access the same state data.

By default, Terraform implicitly uses a backend called local to store state as a local file on disk. Every other backend stores state in a remote service of some kind, which allows multiple people to access it. Accessing state in a remote service generally requires some kind of access credentials, since state data contains extremely sensitive information.

Some backends act like plain "remote disks" for state files; others support locking the state while operations are being performed, which help prevent conflicts and inconsistencies.

upvoted 4 times

👤 **stalk98** 2 years, 1 month ago

answer is B

upvoted 2 times

👤 **bigboi23** 2 years, 1 month ago

Selected Answer: B

Some backends act like plain "remote disks" for state files; others support locking the state while operations are being performed, which help prevent conflicts and inconsistencies.

upvoted 2 times

How can terraform plan aid in the development process?

- A. Validates your expectations against the execution plan without permanently modifying state
- B. Initializes your working directory containing your Terraform configuration files
- C. Formats your Terraform configuration files
- D. Reconciles Terraform's state against deployed resources and permanently modifies state using the current status of deployed resources

Correct Answer: A

Reference:

<https://github.com/hashicorp/terraform/issues/19235>

Community vote distribution

A (86%)

14%

 **Eltooth** Highly Voted 2 years ago

Selected Answer: A

A is correct answer : Validates your expectations against the execution plan without permanently modifying state

"The terraform plan command creates an execution plan, which lets you preview the changes that Terraform plans to make to your infrastructure. By default, when Terraform creates a plan it:
Reads the current state of any already-existing remote objects to make sure that the Terraform state is up-to-date.
Compares the current configuration to the prior state and noting any differences.
Proposes a set of change actions that should, if applied, make the remote objects match the configuration."

"The plan command alone will not actually carry out the proposed changes, and so you can use this command to check whether the proposed changes match what you expected before you apply the changes or share your changes with your team for broader review.

If Terraform detects that no changes are needed to resource instances or to root module output values, terraform plan will report that no action need to be taken."

<https://www.terraform.io/cli/commands/plan>

upvoted 8 times

 **Molly1994** Most Recent 1 month, 1 week ago

A is right. D is wrong. terraform plan did not change anything. It only give you an execution plan that shows your changes.

upvoted 1 times

 **rga91** 9 months, 3 weeks ago

Selected Answer: D

D. When you execute a Terraform Plan, a refresh is produced and the state is modified.

Easy example: you have created a Virtual Machine using Terraform and then you manually deleted using for example the AWS console. In the state, it will appear that Virtual Machine exists, but after running Terraform plan, it will detect no longer exists, and it will update the state.

You can try it yourself. It is a very simple test.

PD: the only way to avoid this refresh, is using the flag -refresh=false in terraform plan, but since it does not say anything about it, this is not a option

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

AAAAAAAAAAAAAA

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

Selected Answer: A

Its A, no question.

upvoted 4 times

⊕  **chimons** 1 year, 6 months ago

Selected Answer: A

A. Validates your expectations against the execution plan without permanently modifying state

Terraform's plan command is used to create an execution plan that outlines the steps that Terraform will take to reach your desired infrastructure state. It allows you to preview and validate the changes that will be made to your infrastructure before actually making those changes. This can be helpful in the development process because it allows you to see exactly what will be changed and ensure that it aligns with your expectations before you apply those changes.

upvoted 3 times

⊕  **lxgywil** 1 year, 7 months ago

Selected Answer: D

It's D.

`terraform plan` reads the current state of any already-existing remote objects to MAKE SURE that the Terraform state is up-to-date.

`-refresh-only` - creates a plan whose goal is ONLY to update the Terraform state and any root module output values to match changes made to remote objects outside of Terraform.

`-refresh=false` - disables the DEFAULT behavior of synchronizing the Terraform state with remote objects before checking for configuration changes.

Reference: <https://developer.hashicorp.com/terraform/cli/commands/plan>

upvoted 2 times

⊕  **raf314** 1 year, 2 months ago

Your answer (D) is wrong... Terraform plan does not modify the state plan, which is what D states.

upvoted 2 times

⊕  **MadMarc** 1 year, 4 months ago

D. Reconciles Terraform's state against deployed resources and permanently modifies state using the current status of deployed resources.

Terraform plan does not, by any means, modify the state file permanently. You could run tf plan a hundred times and it will show you the same differences.

The correct answer, and the only logical answer possible, is A.

upvoted 2 times

⊕  **eduvar4** 1 year, 9 months ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/cli/commands/plan>

upvoted 2 times

⊕  **tf_user** 1 year, 10 months ago

Selected Answer: A

A for sure

upvoted 1 times

⊕  **Ahmad_Terraform** 2 years ago

Validates your expectations

upvoted 1 times

⊕  **CHRIS12722222** 2 years ago

A looks good

upvoted 1 times

⊕  **Zam88** 2 years ago

A correct.

upvoted 1 times

You would like to reuse the same Terraform configuration for your development and production environments with a different state file for each. Which command would you use?

- A. terraform import
- B. terraform workspace
- C. terraform state
- D. terraform init

Correct Answer: B

Community vote distribution

B (100%)

 **amrith501** Highly Voted 2 years, 1 month ago

Selected Answer: B

Workspace is the right answer
upvoted 8 times

 **Ahmad_Terraform** Highly Voted 2 years ago

workspace = different state file
upvoted 6 times

 **Ni33** Most Recent 1 year, 2 months ago

Selected Answer: B

BBBBBBBBBB
upvoted 1 times

 **eduvar4** 1 year, 9 months ago

Selected Answer: B

<https://developer.hashicorp.com/terraform/language/state/workspaces#when-to-use-multiple-workspaces>
upvoted 2 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer : Workspaces.

<https://www.terraform.io/language/state/workspaces#when-to-use-multiple-workspaces>
upvoted 4 times

What is the name assigned by Terraform to reference this resource?

```
mainresource "google_compute_instance" "main" {  
    name = "test"  
}
```

- A. compute_instance
- B. main
- C. google
- D. teat

Correct Answer: B

Community vote distribution

B (100%)

✉  **Eltooth** Highly Voted 2 years ago

Selected Answer: B

B is correct answer : main

See Topic 1 #Question49 for similar question.

upvoted 6 times

✉  **TigerInTheCloud** Most Recent 6 months, 4 weeks ago

invalid terraform config :-)

upvoted 1 times

✉  **IK912** 1 year ago

B is correct

upvoted 1 times

✉  **Power123** 1 year, 3 months ago

Main . B is correct

upvoted 1 times

✉  **zecch** 1 year, 4 months ago

Selected Answer: B

answer is B. Main is the name of resource block.

upvoted 1 times

✉  **Ahmad_Terraform** 2 years ago

Main

is the name of the resources

upvoted 3 times

You're building a CI/CD (continuous integration/ continuous delivery) pipeline and need to inject sensitive variables into your Terraform run. How can you do this safely?

- A. Pass variables to Terraform with a `-var` flag
- B. Copy the sensitive variables into your Terraform code
- C. Store the sensitive variables in a `secure_vars.tf` file
- D. Store the sensitive variables as plain text in a source code repository

Correct Answer: B

Community vote distribution

A (100%)

✉️  **Bere** Highly Voted 11 months ago

Selected Answer: A

The safest method to inject sensitive variables into your Terraform run in a CI/CD pipeline is:

- A. Pass variables to Terraform with a `-var` flag.

When running Terraform in your CI/CD pipeline, you would pass the sensitive variables like this:

```
terraform apply -var="database_password=mysecretpassword" -var="api_key=1234567890abcdef"
```

For enhanced security, in a CI/CD environment, you'd typically use environment variables or secret management tools that are supported by your CI/CD platform to pass these sensitive values, so they are never exposed in logs or stored in insecure locations.

For instance, using environment variables:

```
export TF_VAR_database_password=mysecretpassword
export TF_VAR_api_key=1234567890abcdef
terraform apply
upvoted 11 times
```

✉️  **rfd** Highly Voted 2 years, 1 month ago

Selected Answer: A

Definitely A.

upvoted 7 times

✉️  **samimshaikh** Most Recent 6 months, 2 weeks ago

Selected Answer: A

The safest option among the choices provided is to use the `-var` flag to pass sensitive variables to Terraform. This allows you to provide variable values at runtime without hardcoding them directly into the Terraform code or exposing them in the source code repository.

upvoted 4 times

✉️  **Tlakmini** 11 months, 1 week ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/language/values/variables>

upvoted 2 times

✉️  **cloudznej** 1 year ago

I see that is A but since its on B. Will it be B on the exam? Or that option, even though everyone is on A. Just curious.

upvoted 1 times

✉️  **LunarPhobia** 11 months, 3 weeks ago

The answers on here aren't always correct. Typically go with the highest rated one

upvoted 2 times

✉️  **Faaizz** 1 year, 3 months ago

Selected Answer: A

Obviously A

upvoted 2 times

 **camps** 1 year, 3 months ago

Selected Answer: A

A. Pass variables to Terraform with a "var" flag.

When running Terraform in a CI/CD pipeline, it is important to securely pass sensitive variables to Terraform. One way to pass sensitive variables to Terraform is to use the -var flag to supply the value of the variable at runtime.

upvoted 3 times

 **Power123** 1 year, 3 months ago

A is correct

upvoted 1 times

 **thor7** 1 year, 3 months ago

Selected Answer: A

A is a correct answer

Reference: <https://developer.hashicorp.com/terraform/language/values/variables>

upvoted 1 times

 **eduvar4** 1 year, 9 months ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/language/values/variables>

upvoted 1 times

 **tf_user** 1 year, 10 months ago

Selected Answer: A

A is correct

upvoted 1 times

 **Tomcru1234589** 1 year, 11 months ago

A for sure

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

A= correct

Var

upvoted 1 times

 **Eltooth** 2 years ago

Selected Answer: A

A is correct answer : -var

See Topic 1 # Question 43

<https://blog.gruntwork.io/a-comprehensive-guide-to-managing-secrets-in-your-terraform-code-1d586955ace1>

upvoted 3 times

 **bp39** 2 years, 1 month ago

Selected Answer: A

A is the answer

upvoted 2 times

 **biscuithammer** 2 years, 1 month ago

Selected Answer: A

I vote A

upvoted 1 times

 **bigboi23** 2 years, 1 month ago

Selected Answer: A

I too would go with A

upvoted 1 times

Question #71

Topic 1

Your security team scanned some Terraform workspaces and found secrets stored in a plaintext in state files.

How can you protect sensitive data stored in Terraform state files?

- A. Delete the state file every time you run Terraform
- B. Store the state in an encrypted backend
- C. Edit your state file to scrub out the sensitive data
- D. Always store your secrets in a secrets.tfvars file.

Correct Answer: B

Reference:

<https://www.terraform.io/docs/language/state/sensitive-data.html>

Storing state remotely can provide better security. As of Terraform 0.9, Terraform does not persist state to the local disk when remote state is in use, and some backends can be configured to encrypt the state data at rest.

For example:

- **Terraform Cloud** always encrypts state at rest and protects it with TLS in transit. Terraform Cloud also knows the identity of the user requesting state and maintains a history of state changes. This can be used to control access and track activity. **Terraform Enterprise** also supports detailed audit logging.
- The S3 backend supports encryption at rest when the `encrypt` option is enabled. IAM policies and logging can be used to identify any invalid access. Requests for the state go over a TLS connection.

Community vote distribution

B (83%)

D (17%)

✉  **NashP** 5 months, 2 weeks ago

- B. Store the state in an encrypted backend

Explanation:

Storing the Terraform state in an encrypted backend is the recommended approach to protect sensitive data. Terraform supports various backends, and using an encrypted backend ensures that the state file is encrypted at rest, adding an extra layer of security.

upvoted 1 times

✉  **shefulacertificari** 5 months, 4 weeks ago

Selected Answer: B

B is the answer.

upvoted 1 times

✉  **gofavad926** 9 months, 2 weeks ago

Selected Answer: B

B, got answer:

To protect sensitive data stored in Terraform state files, you should use an encrypted backend. Storing state in an encrypted backend helps secure sensitive information, such as secrets and access keys, that might be present in the state file. This prevents unauthorized access to sensitive data.

- D. Always store your secrets in a secrets.tfvars file: Storing secrets in separate variable files is a common practice, but it doesn't directly add the security of the Terraform state. Even if secrets are in a separate file, protecting the state file is still important.

upvoted 2 times

 **aanataliya** 10 months, 1 week ago

Selected Answer: B

Confusion between B and D? Check this.

Terraform will still record sensitive values in the state, and so anyone who can access the state data will have access to the sensitive values in cleartext

Ref: <https://developer.hashicorp.com/terraform/language/values/variables#suppressing-values-in-cli-output>

Variables in any file is still stored in state as plaintext. So D cannot be correct answer. Please vote for correct answer to help others.

upvoted 2 times

 **Ni33** 1 year, 2 months ago

Selected Answer: D

Why not D? Setting values with a .tfvars file allows you to separate sensitive values from the rest of your variable values, and makes it clear to people working with your configuration which values are sensitive. However, it requires that you maintain and share the secret.tfvars file with the appropriate people. You must also be careful not to check .tfvars files with sensitive values into version control.

upvoted 3 times

 **DKwork** 1 year ago

It cannot be D because the problem is that the secrets are plaintext within your secrets.tfvars file. Think about how you would secure that secrets.tfvars within your group of appropriate people

upvoted 3 times

 **Chrisler** 10 months, 1 week ago

I disagree, setting values with a .tfvar file is also an option but least recommended. Securing your state file by encryption would only give access to the person or a team that needs it.

upvoted 1 times

 **joyboy23** 1 year ago

Wouldn't it still be rendered as plain text in your state files ?

upvoted 2 times

 **camps** 1 year, 3 months ago

Selected Answer: B

B. Store the state in an encrypted backend.

Terraform state files can contain sensitive information such as access keys, passwords, and private keys. To protect this information from unauthorized access, it is important to store the state file securely.

upvoted 3 times

 **Power123** 1 year, 3 months ago

B is correct

upvoted 1 times

 **vadeemkaa** 1 year, 6 months ago

Definitely B

upvoted 1 times

 **nakikoo** 1 year, 7 months ago

Selected Answer: B

agreeeee

upvoted 1 times

 **eduvar4** 1 year, 9 months ago

Selected Answer: B

<https://developer.hashicorp.com/terraform/language/state/sensitive-data>

upvoted 3 times

 **Ahmad_Terraform** 2 years ago

B

store in encrypted backend , E.g S3

upvoted 3 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer : Store in encrypted backend.

upvoted 3 times

Question #72

Topic 1

In contrast to Terraform Open Source, when working with Terraform Enterprise and Cloud Workspaces, conceptually you could think about them as completely separate working directories.

- A. True
- B. False

Correct Answer: A

Community vote distribution

A (100%)

 **Eltooth** Highly Voted 2 years ago

Selected Answer: A

Very poorly worded question however I believe they are referring to the difference between OSS (directories) and Cloud/Ent that uses workspaces which includes separate working directories.

<https://www.terraform.io/cloud-docs/workspaces>

upvoted 16 times

 **Eltooth** 2 years ago

"When run locally, Terraform manages each collection of infrastructure with a persistent working directory, which contains a configuration, state data, and variables. Since Terraform CLI uses content from the directory it runs in, you can organize infrastructure resources into meaningful groups by keeping their configurations in separate directories."

Terraform Cloud manages infrastructure collections with workspaces instead of directories. A workspace contains everything Terraform needs to manage a given collection of infrastructure, and separate workspaces function like completely separate working directories."

upvoted 8 times

 **Power123** Most Recent 1 year, 3 months ago

A is correct

upvoted 1 times

 **kopper2019** 1 year, 11 months ago

Selected Answer: A

A, True

Question #73

Topic 1

You want to know from which paths Terraform is loading providers referenced in your Terraform configuration (*.tf files). You need to enable debug messages to find this out.

Which of the following would achieve this?

- A. Set the environment variable TF_LOG=TRACE
- B. Set verbose logging for each provider in your Terraform configuration
- C. Set the environment variable TF_VAR_log=TRACE
- D. Set the environment variable TF_LOG_PATH

Correct Answer: A

Reference:

<https://www.terraform.io/docs/cli/config/environment-variables.html>

Terraform refers to a number of environment variables to customize various aspects of its behavior. None of these environment variables are required when using Terraform, but they can be used to change some of Terraform's default behaviors in unusual situations, or to increase output verbosity for debugging.

TF_LOG

Enables detailed logs to appear on stderr which is useful for debugging. For example:

```
export TF_LOG=trace
```

Copy 

Community vote distribution

A (100%)

 **Eltooth** Highly Voted  2 years ago

Selected Answer: A

A is correct answer. TF_LOG=TRACE

Although this will only output to stderr and if you need to review log file you will need to include TF_LOG_PATH=pathoffile

<https://www.terraform.io/internals/debugging>

upvoted 10 times

 **yufei** Highly Voted  2 years ago

A. TRACE is the most verbose option available and should show what the user is looking for.
TRACE will show TRACE, DEBUG, INFO, WARN and ERROR messages in the output.

upvoted 6 times

 **Power123** Most Recent  1 year, 3 months ago

TF_LOG=TRACE is correct. Ans A

upvoted 1 times

 **chimons** 1 year, 6 months ago

Selected Answer: A

A. Set the environment variable TF_LOG=TRACE

To enable debug messages in Terraform, you can set the environment variable TF_LOG to the value TRACE. This will cause Terraform to print detailed debug information, including the paths from which it is loading providers.

upvoted 3 times

 **Ahmad_Terraform** 2 years ago

Set the environment variable TF_LOG=TRACE

upvoted 3 times

Question #74

Topic 1

How is terraform import run?

A. As a part of terraform init

- B. As a part of terraform plan
- C. As a part of terraform refresh
- D. By an explicit call
- E. All of the above

Correct Answer: D

Community vote distribution

D (100%)

✉ Eltooth Highly Voted 2 years ago

Selected Answer: D

I'm going with D based on this doc.

"The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version Terraform will also generate configuration.

Because of this, prior to running terraform import it is necessary to write manually a resource configuration block for the resource, to which the imported object will be mapped.

While this may seem tedious, it still gives Terraform users an avenue for importing existing resources."

<https://www.terraform.io/cli/import/usage>

upvoted 7 times

✉ Eltooth 2 years ago

<https://www.terraform.io/cli/commands/import>
Usage: terraform import [options] ADDRESS ID

Import will find the existing resource from ID and import it into your Terraform state at the given ADDRESS.

ADDRESS must be a valid resource address. Because any resource address is valid, the import command can import resources into modules as well as directly into the root of your state.

ID is dependent on the resource type being imported. If the ID is invalid, you'll just receive an error message.

Warning: Terraform expects that each remote object it is managing will be bound to only one resource address, which is normally guaranteed by Terraform itself having created all objects. If you import existing objects into Terraform, be careful to import each remote object to only one Terraform resource address. If you import the same object multiple times, Terraform may exhibit unwanted behavior.

upvoted 1 times

✉ Clapton79 Most Recent 8 months, 1 week ago

Selected Answer: D

Latest version of Terraform can import as part of apply in case you have import {} clauses.
Since this is not part of the answers, explicit call remains, answer D.

upvoted 1 times

✉ Power123 1 year, 3 months ago

Correct answer is D

upvoted 1 times

✉ Optimus 1 year, 10 months ago

Selected Answer: D

D is correct. I have tested it.
upvoted 2 times

✉ Ahmad_Terraform 2 years ago

By an explicit call

upvoted 3 times

You have a simple Terraform configuration containing one virtual machine (VM) in a cloud provider. You run `terraform apply` and the VM is created successfully.

What will happen if you delete the VM using the cloud provider console, and run `terraform apply` again without changing any Terraform code?

- A. Terraform will remove the VM from state file
- B. Terraform will report an error
- C. Terraform will not make any changes
- D. Terraform will recreate the VM

Correct Answer: D

Community vote distribution

D (85%)

C (15%)

 **vitasac** Highly Voted 2 years, 2 months ago

Selected Answer: D

for sure response D

upvoted 17 times

 **Ipergorta** Highly Voted 2 years, 2 months ago

The answer is D

upvoted 6 times

 **chaoscreator** Most Recent 1 month, 2 weeks ago

The people who answered C are just doomed to fail this exam. This is such a basic foundational knowledge and it's the very core of Terraform and you can't even get this simple one right? This is such a freebie question.

upvoted 2 times

 **vibzr2023** 3 months, 2 weeks ago

D. Terraform will recreate the VM

When you delete a resource like a VM directly through the cloud provider's console (outside of Terraform), the Terraform state file still believes the resource exists, as it's unaware of any changes made outside its management. The next time you run `terraform apply`, Terraform compares the desired state (defined in your Terraform configuration) with the actual state (as recorded in the state file and observed in the cloud environment).

Since the actual VM no longer exists but your Terraform configuration still defines it, Terraform detects this discrepancy and takes action to reconcile the difference by creating a new VM to match the desired state defined in your Terraform configuration. Terraform's goal is always to make the real-world infrastructure match the configuration.

upvoted 1 times

 **Tyler2023** 8 months, 2 weeks ago

I tried this,

I created the storage account first by running `terraform apply`

Then I manually deleted the storage account through azure portal

Then reran the `terraform apply`

- It refreshes the state, which detects that the storage account was gone

- Then it re-creates storage account with the same name but without the data from previous instance

upvoted 5 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: D

D, the instance will be recreated

upvoted 1 times

✉️  **Halimb** 10 months, 2 weeks ago

Selected Answer: D

D. Stop messing around.

This behavior applies to any resource created and managed by Terraform, not just virtual machines, e.g. simple resources like resource group: Azure (like I've tested with) or any other cloud provider's resources.

upvoted 1 times

✉️  **Bluemoon22** 1 year, 2 months ago

answer is D

upvoted 1 times

✉️  **lucasbg** 1 year, 3 months ago

Selected Answer: C

it's C for sure. Since the VM was deleted using the cloud web/console, the tfstate will still contain the VM information there, hence nothing will be done. This is drift, and the steps here will be to run terraform -apply | plan -refresh-only, the tfstate will update that the VM was removed and then you will run apply again.

upvoted 3 times

✉️  **Mridul31792** 1 year, 2 months ago

Correct answer is D as terraform apply implicitly runs terraform refresh before applying the changes which will clear the data of VM from the state file and plan will show to create a new VM.

upvoted 2 times

✉️  **Power123** 1 year, 3 months ago

D is correct

upvoted 1 times

✉️  **robertninho** 1 year, 6 months ago

Correct answer is D. Terraform will recreate the VM.

In Terraform, the state file is used to store the current state of your infrastructure. When you run terraform apply, Terraform compares the state of your infrastructure as defined in the configuration files with the state recorded in the state file, and then makes any necessary changes to bring the infrastructure into compliance with the configuration.

upvoted 4 times

✉️  **vadeemkaa** 1 year, 6 months ago

Definitely the answer is D

upvoted 1 times

✉️  **chimons** 1 year, 6 months ago

Selected Answer: D

A refresh will be made before apply, therefore terraform will detect the VM is missing, and will update state accordingly. Then, it will create a new one to match configuration

upvoted 5 times

✉️  **adouban** 1 year, 7 months ago

Selected Answer: D

I have tested this on my lab on oracle cloud,
1- created a VM using TF
2- Deleted VM manually
3- on Terraform apply, attempted to recreate the instance

D is the correct answer

upvoted 3 times

✉️  **Raghav_123** 1 year, 7 months ago

The ans is C

upvoted 1 times

✉  **secdaddy** 1 year, 7 months ago

Not sure how you can get C

I just did this :

1. created a VM on AWS using terraform apply
 2. used the AWS console to delete the VM
 3. ran terraform apply again without changing any Terraform code
- Result : terraform recreated the VM (D)

upvoted 3 times

✉  **GHOST1985** 1 year, 7 months ago

Selected Answer: C

Of course C

Question #76

Topic 1

Which of these options is the most secure place to store secrets for connecting to a Terraform remote backend?

- A. Defined in Environment variables
- B. Inside the backend block within the Terraform configuration
- C. Defined in a connection configuration outside of Terraform
- D. None of above

Correct Answer: A

Community vote distribution

C (53%)

A (45%)

2%

✉  **zyxphreez** Highly Voted 1 year, 10 months ago

Selected Answer: A

Definitely is: A

<https://www.terraform.io/language/settings/backends/configuration#credentials-and-sensitive-data>

Warning: We recommend using environment variables to supply credentials and other sensitive data. If you use -backend-config or hardcode these values directly in your configuration, Terraform will include these values in both the .terraform subdirectory and in plan files. This can leak sensitive credentials.

upvoted 26 times

✉  **Alandt** 5 months, 3 weeks ago

Authentication outside of Terraform is more secure than environment variables. Your environment variables can still refer to a file or the definition of your variables inside terraform. So I would go for C.

upvoted 2 times

✉  **deepeshukla** 1 year ago

I will select C. In option A, any debugging will still disclose data.

upvoted 1 times

✉  **Gomjaba** 10 months, 1 week ago

I presume they are hinting at vault here.

upvoted 1 times

✉  **CHRIS12722222** Highly Voted 1 year, 11 months ago

Selected Answer: C

I will go for option C. Whenever possible, it is best to authenticate outside of terraform to keep secrets out of state file

upvoted 15 times

✉  **Alandt** 5 months, 3 weeks ago

I agree with this.

upvoted 1 times

 **SureNot** Most Recent 1 month ago

Selected Answer: C

Let's imagine use AWS S3 bas a backend. Credentials to S3 Bucket are stored in `~/.aws/credentials` file - Outside of terraform, most secure way
upvoted 1 times

 **Molly1994** 1 month, 1 week ago

C vault as example
upvoted 1 times

 **deepakpamban** 2 months, 1 week ago

Option C
upvoted 2 times

 **Venki_dev** 2 months, 3 weeks ago

Selected Answer: C

C. Defined in a connection configuration outside of Terraform (Most Secure)

This is the most secure option. Here, you store your secrets in a separate dedicated location outside of your Terraform configuration. There are several ways to achieve this:

Secret Management Tools: Utilize tools like HashiCorp Vault, AWS Secrets Manager, or Azure Key Vault to store and manage your secrets securely. These tools offer access control and encryption mechanisms.

Encrypted Files: Store secrets in an encrypted file outside your Terraform configuration directory. Terraform can access these secrets during execution by referencing the decrypted content of the file.

upvoted 4 times

 **kingfighters** 3 months, 1 week ago

choose A:

when we use vault, we still need to download it into a file, here is official doc:

- **File**: A configuration file may be specified via the `init` command line. To specify a file, use the `-backend-config=PATH` option when running `terraform init`. If the file contains secrets it may be kept in a secure data store, such as [Vault](<https://www.vaultproject.io/>), in which case it must be downloaded to the local disk before running Terraform.

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#credentials-and-sensitive-data>
upvoted 1 times

 **aksliveswithaws** 3 months, 2 weeks ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#credentials-and-sensitive-data>:~:text=and%20apply%20steps.,-backend%20types,-The%20block%20label
upvoted 1 times

 **AntonyPeter7** 4 months, 2 weeks ago

Selected Answer: C

Authentication outside of Terraform is more secure than environment variables. Like using terraform vault or cloud
upvoted 1 times

 **Kaname93** 4 months, 2 weeks ago

Selected Answer: A

From the documentation :

Warning: We recommend using environment variables to supply credentials and other sensitive data. If you use `-backend-config` or hardcode these values directly in your configuration, Terraform will include these values in both the `.terraform` subdirectory and in plan files. This can lead to sensitive credentials.

So it's A

upvoted 1 times

 **Alandt** 5 months, 3 weeks ago

Selected Answer: C

Definitely C. Authentication outside of Terraform is the most secure way.

upvoted 1 times

enook 5 months, 4 weeks ago

Selected Answer: C

Chat GPT: The most secure option for storing secrets for connecting to a Terraform remote backend is typically:

C. Defined in a connection configuration outside of Terraform

Storing sensitive information, such as authentication credentials, outside of the Terraform configuration helps enhance security by preventing accidental exposure or leakage of sensitive data. Using external tools or configuration management systems to manage secrets can provide additional layers of security and access control. It is generally not recommended to store sensitive information directly within the Terraform configuration (option B) to minimize the risk of inadvertent exposure. Additionally, environment variables (option A) can be a good practice for storing secrets securely, but they need to be managed carefully to avoid unintended exposure.

upvoted 2 times

parag09 6 months ago

Selected Answer: A

The most secure place to store secrets for connecting to a Terraform remote backend is typically defined in environment variables.

upvoted 1 times

vipulchoubisa 6 months, 1 week ago

Warning: We recommend using environment variables to supply credentials and other sensitive data. If you use -backend-config or hardcode these values directly in your configuration, Terraform will include these values in both the .terraform subdirectory and in plan files. This can leak sensitive credentials.

ANSWER SHOULD BE "A"

upvoted 1 times

samimshaikh 6 months, 2 weeks ago

Selected Answer: C

C. Defined in a connection configuration outside of Terraform

The most secure option for storing secrets for connecting to a Terraform remote backend is to define them in a connection configuration outside of Terraform. This involves using external configuration files or secure credential management tools.

Option A (defined in environment variables) is also a good practice for sensitive information, but it might be less secure than an external configuration file if, for example, there is a risk of exposing environment variables.

Option B (inside the backend block within the Terraform configuration) is generally not recommended for storing sensitive information like secrets because Terraform configuration files may be versioned and shared, posing a security risk.

Therefore, when dealing with sensitive information, it's a good practice to use external and secure methods for configuration, such as a separate configuration file or a secure credential management tool.

upvoted 3 times

[Removed] 6 months, 4 weeks ago

Selected Answer: D

It seems to be D

upvoted 1 times

TigerInTheCloud 6 months, 4 weeks ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration>

Warning: We recommend using environment variables to supply credentials and other sensitive data. If you use -backend-config or hardcode these values directly in your configuration, Terraform will include these values in both the .terraform subdirectory and in plan files. This can leak sensitive credentials.

upvoted 1 times

Your DevOps team is currently using the local backend for your Terraform configuration. You would like to move to a remote backend to begin storing the state file in a central location.

Which of the following backends would not work?

- A. Amazon S3
- B. Artifactory
- C. Git
- D. Terraform Cloud

Correct Answer: A

Reference:

<https://secrethub.io/blog/secret-management-for-terraform/>

Community vote distribution

C (62%)

B (38%)

✉️  **Reet** Highly Voted 11 months, 2 weeks ago

Selected Answer: B

Note: We removed the artifactory, etcd, etcdv3, manta, and swift backends in Terraform v1.3. Information about their behavior in older version still available in the Terraform v1.2 documentation. For migration paths from these removed backends, refer to Upgrading to Terraform v1.3.
<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#available-backends>

upvoted 13 times

✉️  **ray320x** 1 month, 2 weeks ago

With the removal of support for Artifactory in Terraform v1.3 and Git never being a supported backend, both options B and C would not work. However, Git has never been a supported backend for Terraform state files, making it the primary correct answer.

The correct answer is:

C. Git

upvoted 1 times

✉️  **Tyler2023** 8 months, 2 weeks ago

Thanks for this, as I question why artifactory

upvoted 2 times

✉️  **nez15** Highly Voted 2 years, 2 months ago

Selected Answer: C

<https://www.terraform.io/cdktf/concepts/remote-backends>

upvoted 8 times

✉️  **ray320x** Most Recent 1 month, 2 weeks ago

With the removal of support for Artifactory in Terraform v1.3 and Git never being a supported backend, both options B and C would not work. However, Git has never been a supported backend for Terraform state files, making it the primary correct answer.

The correct answer is:

C. Git

upvoted 1 times

✉️  **Absence379** 4 months, 1 week ago

B

Note: We removed the artifactory, etcd, etcdv3, manta, and swift backends in Terraform v1.3. Information about their behavior in older version still available in the Terraform v1.2 documentation. For migration paths from these removed backends, refer to Upgrading to Terraform v1.3.

upvoted 1 times

✉️  **KLIERKO** 9 months, 2 weeks ago

Selected Answer: C

The answer is C stop kliering

upvoted 1 times

✉  **gofavad926** 9 months, 2 weeks ago

Selected Answer: C

C. git is not supported

upvoted 1 times

✉  **Bere** 11 months ago

Selected Answer: C

Supported backends:

<https://developer.hashicorp.com/terraform/cdktf/concepts/remote-backends#supported-backends>

Git is not a supported backend for Terraform state. While it's possible to store state files in Git, doing so is not recommended because state files can contain sensitive information, and it's not designed for concurrent state management.

upvoted 3 times

✉  **Tlakmini** 11 months, 1 week ago

Selected Answer: B

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#available-backends>

upvoted 2 times

✉  **sumnear** 11 months, 2 weeks ago

Selected Answer: C

C is the answer

upvoted 1 times

✉  **wheelan** 11 months, 4 weeks ago

Selected Answer: B

Answer is B. I am using Git as a backend.

Artifactory is no longer a valid backend.

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#available-backends>

upvoted 2 times

✉  **kingfighters** 3 months, 1 week ago

git is not official support now, check the doc: <https://developer.hashicorp.com/terraform/language/settings/backends/local>

upvoted 1 times

✉  **cedry87** 1 year, 3 months ago

answer : B

upvoted 1 times

✉  **camps** 1 year, 3 months ago

Selected Answer: C

C. Git.

Git is not a suitable remote backend for Terraform. While it is possible to store Terraform state files in Git, it is not recommended because Git is not designed to manage state files and does not provide the necessary locking mechanisms to prevent concurrent access.

upvoted 2 times

✉  **ArnaldoW** 1 year, 3 months ago

If Everyone knows, that the answer is C why didn't the admin fix it yet?

upvoted 6 times

✉  **kapara** 11 months ago

I believe the administrator is aware of the solution. However, if they merely provide us with answers without encouraging exploration, individuals might resort to memorizing the responses. This could potentially lead to a situation where the company detects and takes action against the website for distributing exam dumps. It's essential for learning and growth that we understand concepts thoroughly through exploration rather than relying solely on pre-supplied answers.

upvoted 12 times

✉  **vadeemkaa** 1 year, 6 months ago

Answer C - Git cannot be used as a backend

upvoted 1 times

✉  **Stiffler1** 1 year, 7 months ago

Why is answer showing A as the correct answer. C is the answer for sure

upvoted 2 times

 **vikramvlr** 1 year, 7 months ago

Answer is C - Git is not supported to do the remote backend operation

upvoted 1 times

 **karapet** 1 year, 8 months ago

"C" for sure.

upvoted 1 times

Question #78

Topic 1

Which backend does the Terraform CLI use by default?

- A. Terraform Cloud
- B. Consul
- C. Remote
- D. Local

Correct Answer: D

Reference:

<https://www.terraform.io/docs/language/settings/backends/configuration.html>

Default Backend

If a configuration includes no backend block, Terraform defaults to using the `local` backend, which stores state as a plain file in the current working directory.

Community vote distribution

D (100%)

 **Eltooth** Highly Voted 2 years ago

Selected Answer: D

D is correct answer : local

"By default, Terraform implicitly uses a backend called local to store state as a local file on disk. Every other backend stores state in a remote service of some kind, which allows multiple people to access it. Accessing state in a remote service generally requires some kind of access credentials, since state data contains extremely sensitive information."

<https://www.terraform.io/language/settings/backends>

upvoted 7 times

 **shehreen** Most Recent 12 months ago

Selected Answer: D

Answer is : D

upvoted 1 times

 **Power123** 1 year, 3 months ago

Answer is D- local

upvoted 1 times

 **SilentMilli** 1 year, 4 months ago

Selected Answer: D

The default backend used by the Terraform CLI is the local backend. This means that the state file is stored on the local file system in a file named "terraform.tfstate".

upvoted 3 times

 **vadeemkaa** 1 year, 6 months ago

Answer D. By default in Local backend

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

D. Local

upvoted 3 times

 **Zam88** 2 years ago

By default, Terraform uses a backend called local, which stores state as a local file on disk. You can also configure one of the built-in backends listed in the documentation sidebar.

D correct

upvoted 4 times

- A. On disk in the /tmp directory
- B. In memory
- C. On disk in the .terraform sub-directory
- D. They are not cached

Correct Answer: C

Reference:

<https://www.terraform.io/docs/language/modules/sources.html>

Community vote distribution

C (100%)

  **Eltooth** Highly Voted 2 years ago

Selected Answer: C

C is correct answer : hidden terraform directory

"A hidden .terraform directory, which Terraform uses to manage cached provider plugins and modules, record which workspace is currently active, and record the last known backend configuration in case it needs to migrate state on the next run. This directory is automatically managed by Terraform, and is created during initialization."

<https://www.terraform.io/cli/init>

upvoted 11 times

  **amrith501** Highly Voted 2 years, 1 month ago

Selected Answer: C

on disk for sure for sure

upvoted 5 times

  **Eltooth** 2 years ago

On disk in the .terraform sub-directory - just to clarify.

upvoted 1 times

  **shefulacertificari** Most Recent 5 months, 3 weeks ago

Answer - C. It's stored in the .terraform directory.

upvoted 1 times

  **Ni33** 1 year, 2 months ago

Selected Answer: C

C is the correct option

upvoted 1 times

  **Power123** 1 year, 3 months ago

Answer is C

upvoted 1 times

  **vadeemkaa** 1 year, 6 months ago

Answer C. On disk in the .terraform sub-directory

upvoted 1 times

  **eduvar4** 1 year, 9 months ago

Selected Answer: C

<https://developer.hashicorp.com/terraform/language/modules/sources>

upvoted 1 times

  **Ahmad_Terraform** 2 years ago

C. On disk in the .terraform sub-directory

upvoted 1 times

Question #80

Topic 1

You write a new Terraform configuration and immediately run `terraform apply` in the CLI using the local backend.

Why will the apply fail?

- A. Terraform needs you to format your code according to best practices first
- B. Terraform needs to install the necessary plugins first
- C. The Terraform CLI needs you to log into Terraform cloud first
- D. Terraform requires you to manually run `terraform plan` first

Correct Answer: C

Community vote distribution

B (100%)

 **nez15** Highly Voted 2 years, 2 months ago

Selected Answer: B

Need to run Terraform Init first to install the plugins
upvoted 25 times

 **camps** Highly Voted 1 year, 3 months ago

Selected Answer: B

B. Terraform needs to install the necessary plugins first.

When running `terraform apply` for a new Terraform configuration, Terraform needs to install the necessary provider plugins and any dependencies before it can create or modify any resources. Without the necessary plugins, Terraform cannot fully understand the resources that are defined in the configuration file.

upvoted 6 times

 **godie44** Most Recent 2 months, 3 weeks ago

Selected Answer: B

Has to run terraform init first

upvoted 1 times

 **shefulacertificari** 5 months, 3 weeks ago

Answer - B. You need to run 'terraform init' command first to initialize the terraform plugins.

upvoted 3 times

 **LunarPhobia** 11 months, 3 weeks ago

Selected Answer: B

B. You need to run init for required plugins

upvoted 3 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

B for sure.

upvoted 2 times

 **Power123** 1 year, 3 months ago

B is correct

upvoted 2 times

 **Rybitska** 1 year, 4 months ago

Selected Answer: B

it's B

upvoted 2 times

 **vadeemkaa** 1 year, 6 months ago

Answer B. terraform init

upvoted 2 times

 **adouban** 1 year, 7 months ago

Selected Answer: B

B is the correct, terraform init to download required plugins

upvoted 1 times

 **karapet** 1 year, 8 months ago

First of all: terraform init

upvoted 2 times

 **kopper2019** 1 year, 11 months ago

it is B

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

init required

Terraform needs to install the necessary plugins first

upvoted 2 times

 **Ahmad_Terraform** 2 years ago

B plugins first

upvoted 2 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer.

You need to run terraform init before running apply.

upvoted 3 times

 **rfd** 2 years, 1 month ago

Selected Answer: B

terraform init needs to be run first so it can download required modules and setup backend.

upvoted 1 times

 **bp339** 2 years, 1 month ago

Selected Answer: B

B for sure

upvoted 2 times

What features stops multiple admins from changing the Terraform state at the same time?

- A. Version control
- B. Backend types
- C. Provider constraints
- D. State locking

Correct Answer: D

Reference:

<https://blog.gruntwork.io/how-to-manage-terraform-state-28f5697e68fa>

2. Locking: Most version control systems do not provide any form of locking that would prevent two team members from running `terraform apply` on the same state file at the same time.

Community vote distribution

D (100%)

 **Eltooth** Highly Voted 2 years ago

Selected Answer: D

D is correct answer.

Somewhat ambiguous question however the key phrase is "feature".

You need a remote backend first with a State Locking feature available to avoid this scenario.

<https://blog.gruntwork.io/how-to-manage-terraform-state-28f5697e68fa>
upvoted 6 times

 **Ni33** Most Recent 1 year, 2 months ago

Selected Answer: D

D is correct option.

upvoted 1 times

 **Power123** 1 year, 3 months ago

Answer is D

upvoted 1 times

 **vadeemkaa** 1 year, 6 months ago

Answer D. state-lock feature

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

D. State locking

upvoted 3 times

 **amrith501** 2 years, 1 month ago

Selected Answer: D

State locking

upvoted 3 times

A fellow developer on your team is asking for some help in refactoring their Terraform code. As part of their application's architecture, they are going to tear down an existing deployment managed by Terraform and deploy new. However, there is a server resource named `aws_instance.ubuntu[1]` they would like to keep to perform some additional analysis.

What command should be used to tell Terraform to no longer manage the resource?

- A. `terraform apply rm aws_instance.ubuntu[1]`
- B. `terraform state rm aws_instance.ubuntu[1]`
- C. `terraform plan rm aws_instance.ubuntu[1]`
- D. `terraform delete aws_instance.ubuntu[1]`

Correct Answer: *B*

Reference:

<https://www.terraform.io/docs/cli/commands/state/rm.html>

Usage

Usage: `terraform state rm [options] ADDRESS...`

Terraform will search the state for any instances matching the given **resource address**, and remove the record of each one so that Terraform will no longer be tracking the corresponding remote objects.

This means that although the objects will still continue to exist in the remote system, a subsequent `terraform plan` will include an action to create a new object for each of the "forgotten" instances. Depending on the constraints imposed by the remote system, creating those objects might fail if their names or other identifiers conflict with the old objects still present.

Community vote distribution

B (100%)

 **Eltooth** Highly Voted 2 years ago

Selected Answer: B

B is correct answer : terraform state rm

"You can use terraform state rm in the less common situation where you wish to remove a binding to an existing remote object without first destroying it, which will effectively make Terraform "forget" the object while it continues to exist in the remote system."

<https://www.terraform.io/cli/commands/state/rm>

upvoted 13 times

 **Tyler2023** Most Recent 8 months, 2 weeks ago

The answer is correct based on this <https://developer.hashicorp.com/terraform/cli/commands/state/rm>

You can use terraform state rm in the less common situation where you wish to remove a binding to an existing remote object without first destroying it, which will effectively make Terraform "forget" the object while it continues to exist in the remote system.

upvoted 2 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

BBBBBBBBBBBB

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: B

B. terraform state rm aws_instance.ubuntu[1]

To tell Terraform to no longer manage a specific resource, you can use the terraform state rm command. This command removes the specified resource from the Terraform state file, effectively telling Terraform that the resource no longer exists.

upvoted 3 times

 **Power123** 1 year, 3 months ago

B is correct. terraform state rm

Question #83

Topic 1

Terraform can only manage resource dependencies if you set them explicitly with the depends_on argument.

A. True

B. False

Correct Answer: A

Reference:

<https://learn.hashicorp.com/tutorials/terraform/dependencies?in=terraform/0-13>

Community vote distribution

B (93%)

7%

 **vitasac** Highly Voted 2 years, 2 months ago

Selected Answer: B

I think it's B for sure

upvoted 12 times

 **gofavad926** 9 months, 2 weeks ago

"I think", and "for sure"... are not compatible :D

upvoted 22 times

 **Ni33** Highly Voted 1 year, 2 months ago

B is the correct answer. Terraform do manage implicit dependencies using the attribute values mentioned in the configuration of the resources

upvoted 7 times

 **gofavad926** Most Recent 9 months, 2 weeks ago

Selected Answer: B

B, false

upvoted 2 times

 **dhez46** 1 year, 3 months ago

Selected Answer: B

<https://developer.hashicorp.com/terraform/tutorials/0-13/dependencies?in=terraform%2F0-13>

Implicit dependencies are the primary way that Terraform understands the relationships between your resources. Sometimes there are dependencies between resources that are not visible to Terraform, however. The depends_on argument is accepted by any resource or module block and accepts a list of resources to create explicit dependencies for.

upvoted 3 times

 **camps** 1 year, 3 months ago

Selected Answer: B

B. False.

Terraform can manage resource dependencies automatically without explicitly setting them using the depends_on argument.

upvoted 3 times

 **Power123** 1 year, 3 months ago

B is correct. Implicit as well

upvoted 2 times

 **resnef** 1 year, 6 months ago

Selected Answer: B

you have implicit as well, therefore B

upvoted 1 times

 **vikramv1r** 1 year, 7 months ago

Answer is B.

Most of the time, Terraform infers dependencies between resources based on the configuration given, so that resources are created and destroyed in the correct order. Occasionally, however, Terraform cannot infer dependencies between different parts of your infrastructure, and will need to create an explicit dependency with the depends_on argument.

<https://developer.hashicorp.com/terraform/tutorials/0-13/dependencies?in=terraform%2F0-13>

upvoted 3 times

 **eduvar4** 1 year, 9 months ago

Answer: B

"Most of the time, Terraform infers dependencies between resources based on the configuration given, so that resources are created and destroyed in the correct order"

-> <https://learn.hashicorp.com/tutorials/terraform/dependencies>

upvoted 2 times

 **j1809** 1 year, 9 months ago

Terraform automatically infers when one resource depends on another by studying the resource attributes used in interpolation expressions.

upvoted 1 times

 **GopinathM** 1 year, 10 months ago

A is correct we need pass explicitly depends_on

upvoted 1 times

 **kopper2019** 1 year, 11 months ago

Selected Answer: B

It is B

upvoted 1 times

 **Ahmad_Terraform** 2 years ago

B. False

implicit = hidden dependencies Trfm can can manage.

upvoted 2 times

 **Ahmad_Terraform** 2 years ago

B is correct

implicit dependencies trfm can manage without depends on command, for example in AWS eip resource will be dependent on the Ec2 creation, so trfm will first create the Ec2 and then eip will be allocated/created. and eip creation will be after the creation of Ec2.

<https://learn.hashicorp.com/tutorials/terraform/dependencies?in=terraform/0-13>

upvoted 1 times

 **Eltooth** 2 years ago

Selected Answer: A

I believe A is the correct answer.

"Use the depends_on meta-argument to handle hidden resource or module dependencies that Terraform cannot automatically infer. You only need to explicitly specify a dependency when a resource or module relies on another resource's behavior but does not access any of that resource's data in its arguments."

https://www.terraform.io/language/meta-arguments/depends_on

upvoted 2 times

 **RVivek** 1 year, 9 months ago

You voted A and the comment is in support of B ?

upvoted 2 times

 **Zam88** 2 years ago

Manage explicit dependencies

Implicit dependencies are the primary way that Terraform understands the relationships between your resources. Sometimes there are dependencies between resources that are not visible to Terraform, however. The depends_on argument is accepted by any resource or module block and accepts a list of resources to create explicit dependencies for.

B

upvoted 1 times

 **Zam88** 2 years ago

I was wrong i have checked it again - Terraform cannot infer dependencies between different parts of your infrastructure, and you will need to create an explicit dependency with the depends_on argument.

A is correct answer

upvoted 2 times

 **rfd** 2 years, 1 month ago

Selected Answer: B

Terraform is smart enough to manage resource dependencies to an extent, without any explicit instructions.

upvoted 3 times

Question #84

Topic 1

A terraform apply can not _____ infrastructure.

A. change

B. destroy

C. provision

D. import

Correct Answer: D

Community vote distribution

D (100%)

- ✉️  **alejandro_torres** Highly Voted 11 months, 2 weeks ago
deprecated question <https://www.hashicorp.com/blog/terraform-1-5-brings-config-driven-import-and-checks>
upvoted 6 times
- ✉️  **SureNot** Most Recent 1 month ago
Terraform v1.5.0 and later supports import blocks...
A bit old question...
upvoted 1 times
- ✉️  **LunarPhobia** 11 months, 3 weeks ago
Selected Answer: D
D as importing requires a separate command from Apply
upvoted 4 times
- ✉️  **iDiddy** 1 year ago
import is an explicit command
upvoted 3 times
- ✉️  **Clapton79** 8 months, 1 week ago
Not any more, you can have an import {} clause so I guess this question is no longer valid.
<https://www.hashicorp.com/blog/terraform-1-5-brings-config-driven-import-and-checks>
upvoted 2 times
- ✉️  **Power123** 1 year, 3 months ago
Answer is D
upvoted 1 times
- ✉️  **vadeemkaa** 1 year, 6 months ago
Answer D. terraform import command
upvoted 1 times
- ✉️  **root_i_am** 1 year, 9 months ago
Selected Answer: D
The answer is D.

While terraform allows you import existing infrastructure, it is not through the apply command.
upvoted 3 times
- ✉️  **dnscloud02** 1 year, 9 months ago
Not sure --> Terraform can import existing infrastructure resources. This functionality allows you take resources you created by some other means and bring them under Terraform management.
upvoted 1 times
- ✉️  **antivrillee** 1 year, 9 months ago
The question is about the command "terraform apply" and not terraform
upvoted 1 times
- ✉️  **antivrillee** 1 year, 10 months ago
Selected Answer: D
<https://www.educative.io/answers/what-is-the-command-to-destroy-infrastructure-in-terraform>
upvoted 3 times

Question #85

Topic 1

You need to constrain the GitHub provider to version 2.1 or greater.

Which of the following should you put into the Terraform 0.12 configuration's provider block?

- A. version >= 2.1
- B. version ~> 2.1
- C. version = >2.1 =>€¤€

D. version = >= 2.1 =<€>

Correct Answer: B

Reference:

<https://github.com/hashicorp/terraform-provider-null/issues/31>

Community vote distribution

D (93%)

7%

✉️  **19kilo** Highly Voted 2 years ago

Selected Answer: D

I found cleaner version of the multiple choices.

- A- version >= 2.1
- B- version ~> 2.1
- C- version = "<= 2.1"
- D- version = ">= 2.1"

The answer is D ">= 2.1". Requires quotes I believe.

upvoted 59 times

✉️  **Chrisler** Highly Voted 10 months, 1 week ago

A- version >= 2.1

This option is not valid in Terraform's provider block. Terraform uses a specific version string format to specify provider versions, and this form is not compatible with the >= operator.

B- version ~> 2.1

This option uses a version constraint that allows Terraform to use any version that is compatible with version 2.1, but it does not enforce using version 2.1 or greater. The ~> operator specifies that it should use a version greater than or equal to 2.1 but less than the next major version.

C- version = "<= 2.1"

This option constrains the provider to versions less than or equal to 2.1, which is the opposite of what you want. It limits Terraform to use versions up to and including 2.1 but not greater.

D- version = ">= 2.1"

This option correctly specifies that Terraform should use the GitHub provider version 2.1 or any version greater than 2.1, which matches your requirement of constraining it to version 2.1 or greater.

upvoted 7 times

✉️  **090200f** Most Recent 1 day, 18 hours ago

version = >= 2.1 =<€>

why option D showing like this? not version = ">=2.1" ? any idea.

even in exam they will give like this lambda exp?

upvoted 1 times

✉️  **smittt** 3 months, 2 weeks ago

in 0.12 terraform version

>=2.1, <3.0.0

in 0.13 or later terraform version

~> 2.1

upvoted 1 times

✉️  **gofavad926** 9 months, 2 weeks ago

Selected Answer: D

D- version = ">= 2.1"

upvoted 2 times

Hp45 10 months, 1 week ago

The votes seem incorrect. Correct answer should be the ~ sign. See the provider block on registry page:-
aws = {
source = "hashicorp/aws"
version = "~> 5.0"
upvoted 2 times

Bere 10 months, 3 weeks ago

Selected Answer: D

- A- version >= 2.1
- B- version ~> 2.1
- C- version = "<= 2.1"
- D- version = ">= 2.1"

Example:

```
provider "github" {  
version = ">= 2.1"  
# Other configurations for the provider  
}
```

upvoted 4 times

Mikhael1984 11 months ago

For 0.12 version (1.1.x and earlier), exactly, refer <https://developer.hashicorp.com/terraform/language/v1.1.x/expressions/version-constraints>.

= (or no operator): Allows only one exact version number. Cannot be combined with other conditions.

!=: Excludes an exact version number.

>, >=, <, <=: Comparisons against a specified version, allowing versions for which the comparison is true. "Greater-than" requests newer versions, and "less-than" requests older versions.

>: Allows only the rightmost version component to increment. For example, to allow new patch releases within a specific minor release, use full version number: ~> 1.0.4 will allow installation of 1.0.5 and 1.0.10 but not 1.1.0. This is usually called the pessimistic constraint operator.

upvoted 1 times

Ni33 1 year, 2 months ago

Selected Answer: D

D is the correct option
upvoted 1 times

kiran15789 1 year, 2 months ago

Selected Answer: D

based on 19kilo cleaner version
upvoted 2 times

camps 1 year, 3 months ago

Selected Answer: D

. version = ">= 2.1"

In Terraform, the provider block is used to configure a provider plugin that Terraform uses to interact with a specific cloud or service. The version argument in the provider block is used to constrain the version of the provider plugin that Terraform should use.

upvoted 2 times

Power123 1 year, 3 months ago

D is correct .version = ">= 2.1"

upvoted 1 times

👤 **Nunyabiznes** 1 year, 3 months ago

Selected Answer: D

To constrain the GitHub provider to version 2.1 or greater in a Terraform 0.12 configuration, you should use the following constraint in the provider block:

D. `version = ">= 2.1"`

This constraint indicates that the provider version must be greater than or equal to 2.1. Here's an example of how to configure this in the provider block:

```
provider "github" {  
  version = ">= 2.1"  
  ...  
}
```

This ensures that the GitHub provider used will be at least version 2.1, allowing for any version greater than or equal to 2.1.
upvoted 3 times

👤 **Ahmed_Elmelegy** 1 year, 4 months ago

Selected Answer: D

= "`>=`"

upvoted 1 times

👤 **agmesas** 1 year, 5 months ago

Selected Answer: D

D. `version = ">= 2.1"` (must be a literal string)

upvoted 2 times

Question #86

Topic 1

You just scaled your VM infrastructure and realized you set the count variable to the wrong value. You correct the value and save your change. What do you do next to make your infrastructure match your configuration?

- A. Run an apply and confirm the planned changes
- B. Inspect your Terraform state because you want to change it
- C. Reinitialize because your configuration has changed
- D. Inspect all Terraform outputs to make sure they are correct

Correct Answer: A

Community vote distribution

A (92%)

8%

👤 **vitasac**  2 years, 2 months ago

Selected Answer: A

I'm agree it's A

upvoted 12 times

👤 **090200f**  1 day, 18 hours ago

A: run an apply is correct ans

upvoted 1 times

👤 **Power123** 1 year, 3 months ago

A is correct

upvoted 1 times

 **RVivek** 1 year, 9 months ago

Selected Answer: A

A-- is required to deploy the the changed count
B- Not required
C-- checked in my Lab for the count change it worked with out init
D Not required
upvoted 4 times

 **Eltooth** 2 years ago

Selected Answer: A

A is correct answer : apply

You need your deployment to match your config, so only way to implement changes is through terraform apply.
upvoted 4 times

 **leonelferrari** 2 years ago

A is correct

upvoted 2 times

 **stalk98** 2 years, 1 month ago

A is the answer
upvoted 1 times

 **wangchung** 2 years, 1 month ago

D? Its like throughout all these questions for this exam the answers are often not just wrong, but the obvious wrong one.

A

upvoted 1 times

 **Sunrayk** 2 years, 2 months ago

It should be A as the plan will show you what will change and then you can confirm the apply
upvoted 3 times

 **Sunrayk** 2 years, 2 months ago

Actually no, it should be C as the config has changed.
<https://www.terraform.io/cli/commands/init>
upvoted 1 times

 **vikramv1r** 1 year, 7 months ago

initialize is used to refer the plug-ins
upvoted 1 times

 **Parthasarathi** 2 years, 2 months ago

Selected Answer: C

It should be C as the configuration file is changed
upvoted 2 times

 **Maaran07** 2 years, 1 month ago

Configuration change don't require init
upvoted 3 times

 **nez15** 2 years, 2 months ago

Selected Answer: A

Not sure why D. Should be A
upvoted 4 times

Terraform provisioners that require authentication can use the _____ block.

- A. connection
- B. credentials
- C. secrets
- D. ssh

Correct Answer: B

Community vote distribution

A (100%)

✉️  BaburTurk Highly Voted 10 months, 3 weeks ago

Selected Answer: A

The answer is A. connection.

The connection block is used to configure the authentication settings for a Terraform provisioner. This includes the username, password, and SSH key that will be used to connect to the remote resource.

The credentials and secrets blocks are used to store sensitive information, such as passwords and SSH keys. These blocks are not directly used by provisioners, but they can be referenced by the connection block.

The ssh block is used to configure the SSH client that Terraform will use to connect to the remote resource. This block is not typically used by provisioners, as the connection block can be used to configure the SSH client for all provisioners.

Here is an example of a connection block for a provisioner that requires SSH authentication:

```
connection {  
host = "example.com"  
user = "root"  
password = "my-password"  
}
```

upvoted 8 times

✉️  NashP Most Recent 5 months, 2 weeks ago

```
resource "aws_instance" "example" {  
ami = "ami-0c55b159cbfafe1f0"  
instance_type = "t2.micro"  
  
connection {  
type = "ssh"  
user = "ec2-user"  
private_key = file("~/ssh/private_key.pem")  
}
```

```
provisioner "remote-exec" {  
inline = [  
"echo 'Hello, World!' > hello.txt",  
]  
}
```

A connection
upvoted 2 times

✉️  Tlakmini 11 months, 1 week ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/cli/config/config-file>

upvoted 1 times

✉  **Power123** 1 year, 3 months ago

A is correct. Connection block
upvoted 2 times

✉  **tgaos** 1 year, 4 months ago

Selected Answer: A

<https://developer.hashicorp.com/terraform/language/resources/provisioners/connection>

Most provisioners require access to the remote resource via SSH or WinRM and expect a nested connection block with details about how to connect.

upvoted 4 times

✉  **resnef** 1 year, 6 months ago

Selected Answer: A

connection block: <https://developer.hashicorp.com/terraform/language/resources/provisioners/connection#connection-block>

upvoted 1 times

✉  **adouban** 1 year, 7 months ago

Selected Answer: A

A is correct

upvoted 1 times

✉  **asudhin** 1 year, 9 months ago

Selected Answer: A

It is A

upvoted 1 times

✉  **Burakko** 1 year, 10 months ago

Selected Answer: A

Must be the connection block.

upvoted 1 times

✉  **aadi1121** 1 year, 11 months ago

Selected Answer: A

<https://www.terraform.io/language/resources/provisioners/connection>

upvoted 3 times

✉  **Eltooth** 2 years ago

Selected Answer: A

A is correct answer : connection.

"Most provisioners require access to the remote resource via SSH or WinRM and expect a nested connection block with details about how to connect."

"Connection blocks don't take a block label and can be nested within either a resource or a provisioner."

<https://www.terraform.io/language/resources/provisioners/connection>

upvoted 3 times

✉  **Zam88** 2 years ago

Most provisioners require access to the remote resource via SSH or WinRM and expect a nested connection block with details about how to connect.

A

upvoted 2 times

Terraform validate reports syntax check errors from which of the following scenarios?

- A. Code contains tabs indentation instead of spaces
- B. There is missing value for a variable
- C. The state files does not match the current infrastructure
- D. None of the above

Correct Answer: B

Reference:

<http://man.hubwiz.com/docset/Terraform.docset/Contents/Resources/Documents/docs/commands/validate.html>

The `terraform validate` command is used to validate the syntax of the terraform files. Terraform performs a syntax check on all the terraform files in the directory, and will display an error if any of the files doesn't validate.

This command **does not** check formatting (e.g. tabs vs spaces, newlines, comments etc.).

The following can be reported:

- invalid HCL syntax (e.g. missing trailing quote or equal sign)
- invalid HCL references (e.g. variable name or attribute which doesn't exist)
- same provider declared multiple times
- same module declared multiple times
- same resource declared multiple times
- invalid module name
- interpolation used in places where it's unsupported (e.g. `variable`, `depends_on`, `module.source`, `provider`)
- missing value for a variable (none of `-var foo=...` flag, `-var-file=foo.vars` flag, `TF_VAR_foo` environment variable, `terraform.tfvars`, or default value in the configuration)

Community vote distribution

D (57%)

B (41%)

2%

 **Eltooth** Highly Voted 2 years ago

Selected Answer: B

B is correct answer.

The terraform validate command is used to validate the syntax of the terraform files. Terraform performs a syntax check on all the terraform files in the directory, and will display an error if any of the files doesn't validate.

This command does not check formatting (e.g. tabs vs spaces, newlines, comments etc.).

The following can be reported:

- invalid HCL syntax (e.g. missing trailing quote or equal sign)
- invalid HCL references (e.g. variable name or attribute which doesn't exist)
- same provider declared multiple times
- same module declared multiple times
- same resource declared multiple times
- invalid module name
- interpolation used in places where it's unsupported (e.g. `variable`, `depends_on`, `module.source`, `provider`)
- missing value for a variable (none of `-var foo=...` flag, `-var-file=foo.vars` flag, `TF_VAR_foo` environment variable, `terraform.tfvars`, or default value in the configuration)

<https://www.typeerror.org/docs/terraform/commands/validate>

<https://learning-ocean.com/tutorials/terraform/terraform-validate>

upvoted 29 times

✉  **Pietjeplukgeluk** 1 year, 5 months ago

Just tested this in lab, and "missing value for a variable (none of -var foo=... flag, -var-file=foo.vars flag, TF_VAR_foo environment variable, terraform.tfvars, or default value in the configuration)" is untrue, it is required to declare the variable type (but a value is not required to pass the "terraform validate") e.g. validates if referred somewhere else :

```
variable "ami_type" {  
  type = string  
}
```

upvoted 4 times

✉  **joyboy23** 1 year ago

I tested the same too. VSCode is detecting it but not terraform validate. From exam point of view, I wonder which one is the correct answer

upvoted 1 times

✉  **deepeshukla** 1 year ago

Agree. D is the correct answer. I tested in Lab. Missing variable value will not be reported in terraform valid, because it can be passed w/ running code (hence not invalid code).

upvoted 4 times

✉  **Tyler2023** 8 months, 2 weeks ago

Agree on this, D is the correct answer

Terraform Validate runs checks that verify whether a configuration is syntactically valid and internally consistent, regardless of any provided variables or existing state.

To verify configuration in the context of a particular run (a particular target workspace, input variable values, etc), use the terraform plan command instead, which includes an implied validation check.

<https://developer.hashicorp.com/terraform/cli/commands/validate>

upvoted 2 times

✉  **aanataliya** Highly Voted 10 months, 1 week ago

Selected Answer: D

Confusion between B and D.

B was correct in older version of terraform. It was correct until v0.12.9 but then it was removed in later versions. See below documentation link: As per latest documentation B is incorrect when we have option D available.

<https://github.com/hashicorp/terraform/blob/v0.12.9/website/docs/commands/validate.html.markdown>

<https://github.com/hashicorp/terraform/blob/v0.12.10/website/docs/commands/validate.html.markdown>

Please vote for correct answer to help others.

upvoted 14 times

✉  **AntonyPeter7** Most Recent 4 months, 2 weeks ago

Selected Answer: D

D is the correct answer. Without value for variable, if we run terraform apply, it will ask for value via CLI.

upvoted 3 times

✉  **erhard** 4 months, 2 weeks ago

Selected Answer: D

<https://developer.hashicorp.com/terraform/cli/commands/validate>

... regardless of any provided variables or existing state

upvoted 1 times

✉  **Alandt** 5 months, 3 weeks ago

Selected Answer: D

Newer version of Terraform does not require you to give up values.

upvoted 2 times

👤 **TigerInTheCloud** 6 months, 4 weeks ago

Selected Answer: D

```
$ cat main.tf
variable "cmd" {}
resource "null_resource" "cmd" {
provisioner "local-exec" {
command = "${var.cmd}"
}
}
$ terraform validate
Success! The configuration is valid.
```

upvoted 1 times

👤 **Ramdi1** 7 months, 3 weeks ago

Selected Answer: B

I think it is B

upvoted 1 times

👤 **ghostGuiggs** 8 months, 2 weeks ago

Selected Answer: B

B is the answer

upvoted 1 times

👤 **MisterROBOT** 8 months, 2 weeks ago

D -

The terraform validate command is used to validate the syntax of the terraform files. Terraform performs a syntax check on all the terraform files in the directory, and will display an error if any of the files doesn't validate.

This command does not check formatting (e.g. tabs vs spaces, newlines, comments etc.)

upvoted 1 times

👤 **MisterROBOT** 8 months, 2 weeks ago

A

terraform validate is a command that validates the configuration files in a directory, referring only to the configuration and not accessing any remote services such as remote state, provider APIs, etc. It runs checks that verify whether a configuration is syntactically valid and internally consistent, regardless of any provided variables or existing state. Therefore, it reports syntax check errors if there are any syntax errors in the code.

Therefore, the correct answer is A. Code contains tabs indentation instead of spaces.

upvoted 1 times

👤 **MauroSoli** 8 months, 3 weeks ago

Also input variable of a root module is a variable.

So if a variable of an used module haven't a default value terraform validate return error.

Answer is B

upvoted 1 times

👤 **Aman726** 9 months ago

Selected Answer: D

D is correct

upvoted 1 times

👤 **Alex1atd** 9 months, 1 week ago

Selected Answer: D

You can have missing value on variable, so D is the answer

upvoted 2 times

👤 **adityanarayan** 9 months, 1 week ago

i have checked practically with none variable value output say its valid so option D

upvoted 1 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: B

- B. There is missing value for a variable

Terraform validate will check for syntax errors in your Terraform configuration, including the following:

Missing or invalid variable values
Invalid resource references
Invalid attribute values
Invalid data source references
Invalid module references

Terraform validate will not check for the following:

Code indentation
State file consistency
Other configuration errors, such as logical errors or errors in the desired state of your infrastructure
upvoted 1 times

 **Halimb** 10 months, 2 weeks ago

Selected Answer: D

- D. None of the above.

Ref: <https://developer.hashicorp.com/terraform/cli/commands/validate>

upvoted 1 times

 **Spandrop** 10 months, 2 weeks ago

D is the correct answer

"Validate runs checks that verify whether a configuration is syntactically valid and internally consistent, regardless of any provided variables or existing state. It is thus primarily useful for general verification of reusable modules, including correctness of attribute names and value types.

<https://developer.hashicorp.com/terraform/cli/commands/validate>

upvoted 1 times

Question #89

Topic 1

Which of the following is allowed as a Terraform variable name?

- A. count
- B. name
- C. source
- D. version

Correct Answer: B

Reference:

<https://www.terraform.io/docs/language/values/variables.html>

The label after the `variable` keyword is a name for the variable, which must be unique among all variables in the same module. This name is used to assign a value to the variable from outside and to reference the variable's value from within the module.

The name of a variable can be any valid `identifier` except the following: `source` , `version` , `providers` , `count` , `for_each` , `lifecycle` , `depends_on` , `locals` .

These names are reserved for meta-arguments in `module configuration blocks`, and cannot be declared as variable names.

Community vote distribution

B (100%)

 **Eltooth** Highly Voted 2 years ago

Selected Answer: B

B is correct answer : name

"The name of a variable can be any valid identifier except the following: source, version, providers, count, for_each, lifecycle, depends_on, locals."

<https://www.terraform.io/language/values/variables>

upvoted 9 times

 **Zam88** Highly Voted 2 years ago

The name of a variable can be any valid identifier except the following: source , version , providers , count , for_each , lifecycle , depends_on , locals . These names are reserved for meta-arguments in module configuration blocks, and cannot be declared as variable names.

B

upvoted 5 times

 **Ni33** Most Recent 1 year, 2 months ago

Selected Answer: B

B is correct one.

upvoted 1 times

 **Power123** 1 year, 3 months ago

B is correct. name

upvoted 1 times

 **kopper2019** 1 year, 11 months ago

Selected Answer: B

name is OK

upvoted 4 times

 **Ahmad_Terraform** 2 years ago

name is right

upvoted 2 times

What type of block is used to construct a collection of nested configuration blocks?

- A. for_each
- B. repeated
- C. nesting
- D. dynamic

Correct Answer: D

Reference:

<https://www.hashicorp.com/blog/hashicorp-terraform-0-12-preview-for-and-for-each>

Dynamic Nested Blocks

Several resource types use nested configuration blocks to define repeatable portions of their configuration. Terraform 0.12 introduces a new construct for dynamically constructing a collection of nested configuration blocks.

For example, the `aws_autoscaling_group` resource type uses nested blocks to declare tags that may or may not be propagated to any created EC2 instances. The example below shows the **Terraform 0.11 and earlier syntax**:

Community vote distribution

D (80%)

A (20%)

✉  **bp339** Highly Voted  2 years, 1 month ago

Selected Answer: D

Dynamic is true...

upvoted 8 times

✉  **biprodatta** Most Recent  1 year ago

Selected Answer: A

dynamic doesn't create nested block, for_each does

upvoted 1 times

✉  **Ni33** 1 year, 2 months ago

Selected Answer: A

A is the correct answer.

upvoted 1 times

✉  **Power123** 1 year, 3 months ago

Dynamic block. D

upvoted 1 times

✉  **Abhijeet2904** 1 year, 5 months ago

Selected Answer: A

A. for_each

In Terraform, you can use the "for_each" block to construct a collection of nested configuration blocks that are repeated based on an input list map data structure. The "for_each" block allows you to define reusable code that can be applied to multiple resources, similar to a loop in a programming language. The block takes a data structure as an argument and creates one block of nested configuration for each element in the data structure, allowing you to manage multiple similar resources in a more streamlined way.

upvoted 2 times

✉  **Nunyabiznes** 1 year, 3 months ago

The dynamic block in Terraform is used to construct nested configuration blocks dynamically based on a collection of values, such as a list map. This block type allows you to create multiple instances of the same resource or module, without having to duplicate the configuration code.

The for_each argument is used with the dynamic block to specify the collection of values to iterate over. The content block inside the dynamic block is used to define the configuration for each iteration.

upvoted 10 times

✉  **eduvar4** 1 year, 9 months ago

Selected Answer: D

<https://developer.hashicorp.com/terraform/language/expressions/dynamic-blocks>

upvoted 4 times

✉  **alen995454** 6 months ago

form page shared above:

"A dynamic block acts much like a for expression, but produces nested blocks instead of a complex typed value. It iterates over a given complex value, and generates a nested block for each element of that complex value."

Question #91

Topic 1

Module variable assignments are inherited from the parent module and do not need to be explicitly set.

A. True

B. False

Correct Answer: B

Reference:

<https://github.com/hashicorp/terraform/issues/15818>

Community vote distribution

B (82%)

A (18%)

 **stalk98** Highly Voted 2 years, 1 month ago

Modules do not inherit variables from the parent module. All modules are self-contained units. So you have to explicitly define variables in the child module, and then explicitly set these variables in the parent module, when you instantiate the child module.

upvoted 18 times

 **Bere** Most Recent 9 months ago

Selected Answer: B

Module variable assignments must be explicitly set by the parent module when it instantiates the child module.

Variable Declaration in Child Module (modules/vm/main.tf):

```
variable "instance_type" {
  type = string
  default = "t2.micro" # This is an optional default value.
}
```

```
resource "aws_instance" "example" {
  instance_type = var.instance_type
  ami           = "ami-abc123"
  subnet_id     = "subnet-1234abcd"
}
```

Variable Assignment in Parent Module (main.tf):

```
module "vm" {
  source = "./modules/vm"

  instance_type = "t3.micro" # Assigning a value to the variable.
}
```

In this example, `instance_type` is declared as a variable in the child module and is assigned a value by the parent module when it instantiates the `vm` module. The value "`t3.micro`" assigned by the parent module will override the default value "`t2.micro`" declared in the child module.

upvoted 3 times

 **modarov** 11 months, 2 weeks ago

The answer is False.

Module variable assignments are not inherited from the parent module and do need to be explicitly set.

upvoted 2 times

 **cracit** 1 year, 1 month ago

Modules do not inherit variables from the parent module. All modules are self-contained units. So you have to explicitly define variables in the child module, and then explicitly set these variables in the parent module, when you instantiate the child module.

upvoted 2 times

 **DineshSG** 1 year, 2 months ago

I think we are confused between variable assignment and variable declaration. Variable need to be declared in child module but not necessarily be assigned. We can pass values from parent module

upvoted 3 times

 **Ni33** 1 year, 2 months ago

Selected Answer: A

I think A is the correct answer. Variable value assignments can be passed from root module variables to child module.

upvoted 2 times

 **Eltooth** 2 years ago

Selected Answer: B

B is correct answer : false.

upvoted 4 times

 **amrith501** 2 years, 1 month ago

Selected Answer: B

Modules do not inherit variables from the parent module

upvoted 2 times

Question #92

Topic 1

If writing Terraform code that adheres to the Terraform style conventions, how would you properly indent each nesting level compared to the one above it?

- A. With four spaces
- B. With a tab
- C. With three spaces
- D. With two spaces

Correct Answer: D

Reference:

<https://www.terraform.io/docs/language/syntax/style.html>

The Terraform parser allows you some flexibility in how you lay out the elements in your configuration files, but the Terraform language also has some idiomatic style conventions which we recommend users always follow for consistency between files and modules written by different teams. Automatic source code formatting tools may apply these conventions automatically.

Note: You can enforce these conventions automatically by running `terraform fmt`.

- Indent two spaces for each nesting level.
- When multiple arguments with single-line values appear on consecutive lines at the same nesting level, align their equals signs:

Community vote distribution

D (92%)

8%

Eltooth Highly Voted 2 years ago

Selected Answer: D

D is correct answer: Indent two spaces for each nesting level.

<https://www.terraform.io/language/syntax/style#style-conventions>

upvoted 12 times

Power123 Most Recent 1 year, 3 months ago

2 spaces. Answer is D

upvoted 1 times

aone 1 year, 3 months ago

The Terraform parser allows you some flexibility in how you lay out the elements in your configuration files, but the Terraform language also has some idiomatic style conventions which we recommend users always follow for consistency between files and modules written by different teams. Automatic source code formatting tools may apply these conventions automatically.

Note: You can enforce these conventions automatically by running `terraform fmt`.

Indent two spaces for each nesting level.

When multiple arguments with single-line values appear on consecutive lines at the same nesting level, align their equals signs:

upvoted 1 times

zecch 1 year, 4 months ago

definitely D

upvoted 1 times

Abhijeet2904 1 year, 5 months ago

Selected Answer: A

A. With four spaces

When writing Terraform code that adheres to the Terraform style conventions, each nesting level should be indented four spaces compared to the level above it. This helps to visually distinguish between different levels of nesting and improves readability of the code.

In Terraform, a common use case for nested blocks is to define resources that depend on other resources. For example, you might have a resource that creates a network security group, and then another resource that creates a virtual machine that references the network security group. In this case, the virtual machine resource would be indented four spaces compared to the network security group resource.

Using four spaces for indentation is a widely accepted standard in the software development community, and it is recommended to use four spaces for indentation in Terraform code as well.

upvoted 1 times

David_C_90 1 year, 4 months ago

don't paste chat gpt answers

upvoted 5 times

alen995454 6 months ago

"Compared to one above" .. not the parent

upvoted 1 times

vadeemkaa 1 year, 6 months ago

Answer D for sure

upvoted 1 times

Which of the following is not an action performed by terraform init?

- A. Create a sample main.tf file
- B. Initialize a configured backend
- C. Retrieve the source code for all referenced modules
- D. Load required provider plugins

Correct Answer: A

Reference:

<https://www.terraform.io/docs/cli/init/index.html>

Community vote distribution

A (100%)

 **Eltooth** Highly Voted 2 years ago

Selected Answer: A

Of options, only A is correct answer : Create a sample main.tf file

- B. Initialize a configured backend : <https://www.terraform.io/cli/commands/init#backend-initialization>
 - C. Retrieve the source code for all referenced modules : <https://www.terraform.io/cli/commands/init#child-module-installation>
 - D. Load required provider plugins : <https://www.terraform.io/cli/commands/init#plugin-installation>
- upvoted 8 times

 **Power123** Most Recent 1 year, 3 months ago

Correct answer is A

upvoted 1 times

 **Nzudin** 1 year, 4 months ago

Selected Answer: A

A of course :')

upvoted 1 times

 **keiffo2** 1 year, 10 months ago

init doen't create an main.tf file.

upvoted 4 times

HashiCorp Configuration Language (HCL) supports user-defined functions.

- A. True
- B. False

Correct Answer: B

Reference:

https://www.packer.io/docs/templates/hcl_templates/functions

The HCL language includes a number of built-in functions that you can call from within expressions to transform and combine values. The general syntax for function calls is a function name followed by comma-separated arguments in parentheses:

```
max(5, 12, 9)
```

[Copy !\[\]\(ef268e2dd8292b32d0233ecef569ec45_img.jpg\)](#)

For more details on syntax, see [Function Calls](#) on the Expressions page.

The HCL language does not support user-defined functions, and so only the functions built in to the language are available for use. The navigation for this section includes a list of all of the available built-in functions.

Community vote distribution

B (100%)

  **kopper2019**  1 year, 11 months ago

Selected Answer: B

False

<https://www.terraform.io/language/functions>

The Terraform language does not support user-defined functions, and so only the functions built into the language are available for use
upvoted 8 times

  **Power123**  1 year, 3 months ago

Answer is B

upvoted 2 times

  **aiya** 1 year, 11 months ago

B - false

The HCL language does not support user-defined functions, and so only the functions built in to the language are available for use.
upvoted 4 times

How can you trigger a run in a Terraform Cloud workspace that is connected to a Version Control System (VCS) repository?

- A. Only Terraform Cloud organization owners can set workspace variables on VCS connected workspaces

- B. Commit a change to the VCS working directory and branch that the Terraform Cloud workspace is connected to
- C. Only members of a VCS organization can open a pull request against repositories that are connected to Terraform Cloud workspaces
- D. Only Terraform Cloud organization owners can approve plans in VCS connected workspaces

Correct Answer: B

Reference:

<https://www.terraform.io/docs/cloud/vcs/index.html>

Terraform Cloud is more powerful when you integrate it with your version control system (VCS) provider. Although you can use many of Terraform Cloud's features without one, a VCS connection provides additional features and improved workflows. In particular:

- When workspaces are linked to a VCS repository, Terraform Cloud can [automatically initiate Terraform runs](#) when changes are committed to the specified branch.
- Terraform Cloud makes code review easier by [automatically predicting](#) how pull requests will affect infrastructure.
- Publishing new versions of a [private Terraform module](#) is as easy as pushing a tag to the module's repository.

We recommend configuring VCS access when first setting up an organization, and you might need to add additional VCS providers later depending on how your organization grows.

Configuring a new VCS provider requires permission to manage VCS settings for the organization. ([More about permissions.](#))

Community vote distribution

B (100%)

 **Eltooth** Highly Voted  2 years ago

Selected Answer: B

B is correct answer.

"In a workspace linked to a VCS repository, runs start automatically when you merge or commit changes to version control.

A workspace is linked to one branch of a VCS repository and ignores changes to other branches. You can specify which files and directories within your repository trigger runs."

<https://www.terraform.io/cloud-docs/run/ui#automatically-starting-runs>

upvoted 14 times

 **modarov** Most Recent  11 months, 2 weeks ago

Question #96

Topic 1

Terraform and Terraform providers must use the same major version number in a single configuration.

A. True

B. False

Correct Answer: B

Reference:

<https://www.terraform.io/docs/language/expressions/version-constraints.html>

Community vote distribution

B (100%)

 **Eltooth** Highly Voted  2 years ago

Selected Answer: B

I think B is correct answer : false.

<https://www.terraform.io/language/expressions/version-constraints#terraform-core-and-provider-versions>

upvoted 10 times

 **Busi57** Most Recent  9 months, 4 weeks ago

Selected Answer: B

I think b

upvoted 1 times

 **Power123** 1 year, 3 months ago

Answer is B. False

upvoted 1 times

 **zecch** 1 year, 4 months ago

Definitely B. terraform version and providers version can be different.

upvoted 3 times

 **nakikoo** 1 year, 6 months ago

Version can be set to what is working with your environment resource

upvoted 1 times

 **Zam88** 2 years ago

B is correct

upvoted 3 times

Question #97

Topic 1

Which statement describes a goal of infrastructure as code?

- A. An abstraction from vendor specific APIs
- B. Write once, run anywhere
- C. A pipeline process to test and deliver software
- D. The programmatic configuration of resources

Correct Answer: D

Community vote distribution

D (84%)

B (16%)

✉  **amrith501** Highly Voted 2 years, 1 month ago

Selected Answer: D

Looks like D

upvoted 10 times

✉  **Eltooth** Highly Voted 2 years ago

Selected Answer: D

D is correct answer.

upvoted 6 times

✉  **modarov** Most Recent 11 months, 2 weeks ago

D. The programmatic configuration of resources.

upvoted 1 times

✉  **Power123** 1 year, 3 months ago

D is correct

upvoted 1 times

✉  **agmesas** 1 year, 5 months ago

Selected Answer: D

A, IaC is not for "specific APIs". B (WORE) an example is JVM or container, no matter where you run your code because there is an abstractio layer like docker, java virtual machine, etc.. where the code run fine. For Terraform you need to write a specific terraform configuration file bas on which cloud you will deploy the infra (we can not deploy a tf file write for AWS in Azure changing only the provider, right?). C, IaC is a part c the pipeline but not the entire pipeline. The correct answer is D, "The programmatic configuration of resources" in a human-readable code.

upvoted 5 times

✉  **mifune** 2 years ago

I think that is B because one of the main features of Terraform is being a "Cloud agnostic" solution which best matches with: "Write once, run anywhere".

upvoted 1 times

✉  **mifune** 2 years ago

Sorry, I misunderstood the question... it's D, true.

upvoted 4 times

✉  **Zam88** 2 years ago

The purpose of infrastructure as code is to enable developers or operations teams to automatically manage, monitor and provision resources, rather than manually configure discrete hardware devices and operating systems. Infrastructure as code is sometimes referred to as programmable or software-defined infrastructure.

D

upvoted 3 times

✉  **stalk98** 2 years, 1 month ago

maybe is B?

upvoted 2 times

⊕  **Roro_Brother** 2 years, 1 month ago

Selected Answer: B

Sure, it's B

upvoted 4 times

⊕  **bigboi23** 2 years, 1 month ago

I think it is B.

The purpose of infrastructure as code is to enable developers or operations teams to automatically manage, monitor and provision resources, rather than manually configure discrete hardware devices and operating systems.

upvoted 2 times

⊕  **tbhttp** 1 year, 11 months ago

So you're argument is supporting Answer D

B states "run anywhere". Like any cloud provider. That is just not the case. Still, migrating to another cloud provider may be less of an headache with IAC.

upvoted 1 times

Question #98

Topic 1

When using Terraform to deploy resources into Azure, which scenarios are true regarding state files? (Choose two.)

- A. When a change is made to the resources via the Azure Cloud Console, the changes are recorded in a new state file
- B. When a change is made to the resources via the Azure Cloud Console, Terraform will update the state file to reflect them during the next plan or apply
- C. When a change is made to the resources via the Azure Cloud Console, the current state file will not be updated
- D. When a change is made to the resources via the Azure Cloud Console, the changes are recorded in the current state file

Correct Answer: AC

Community vote distribution

BC (41%)

C (41%)

Other

⊕  **AzureGurl**  2 years, 1 month ago

IT should be B and C, however the wording is incorrect, Terraform Refresh will only update the state file for new changes via Az Console

Terraform Plan does not change the State file, It will however say the diff in the state.

The Terraform apply will make the changes to the state file to keep it consistent.

So the wording of Option B is bit ambiguous.

upvoted 23 times

⊕  **alirasouli** 1 year, 7 months ago

Option B's wording is too fuzzy.

upvoted 1 times

⊕  **msingh20** 11 months, 2 weeks ago

Terraform plan / apply runs a terraform refresh to update the state with infrastructure so B makes sense.

upvoted 2 times

⊕  **Tyler2023** 8 months, 2 weeks ago

I don't think so, terraform will not generate the configuration changes you've maid manually, terraform apply will remove the configuration you've made that is not included in your tf files

upvoted 1 times

👤 **Nikita74** Highly Voted 1 year, 9 months ago

Selected Answer: C

Only C

upvoted 11 times

👤 **090200f** Most Recent 1 day, 14 hours ago

B, C are correct answers

upvoted 1 times

👤 **vibzr2023** 3 months, 2 weeks ago

I believe only option C is correct. I tested this by spinning up an EC2 instance in AWS using Terraform and then manually updating the "Name" tag in the AWS console. Immediately after that, when I ran terraform plan, it indicated that the tag would be changed from the Terraform-assigned value ("TerraformCI") to the manually applied value ("Rajesh"). This demonstrates that Terraform detects changes made outside of its management and plans to update the infrastructure accordingly.

upvoted 1 times

👤 **vibzr2023** 3 months, 2 weeks ago

terraform plan output

```
# aws_instance.ec2_example will be updated in-place
~ resource "aws_instance" "ec2_example" {
  id          = "i-037feb3cac6a58f05"
  ~ tags       = {
    ~ "Name" = "TerraformCI" -> "Rajesh"
  }
  ~ tags_all   = {
    ~ "Name" = "TerraformCI" -> "Rajesh"
  }
```

terraform apply output

update tag from TerraformCI to Rajesh

upvoted 1 times

👤 **vibzr2023** 3 months, 2 weeks ago

While prevailing sentiment leans towards options B and C, as evidenced by the popularity of ChatGPT's response, I posit that only option C is accurate. Definitive proof of option B's validity hinges on the demonstration of program output reflecting a successful state drift reconciliation.

Here is the chatgpt answer when I demonstrated only C is correct.

Thank you for providing detailed information about your test and findings. Based on your description, it seems there was a misunderstanding regarding how Terraform interacts with external changes.

continued.....

upvoted 1 times

👤 **vibzr2023** 3 months, 2 weeks ago

When you manually changed the "Name" tag in the AWS console and then ran terraform plan, Terraform compared the actual infrastructure state (with the "Name" tag set to "Rajesh") against what's defined in your Terraform configuration (where the tag might be set to "TerraformCI"). Seeing the discrepancy, Terraform planned to update the resource to match the configuration—changing the tag back to "TerraformCI". This behavior is expected and demonstrates Terraform's goal to ensure the infrastructure matches the configuration specified in your .tf files.

continued....

upvoted 1 times

👤 **Absence379** 4 months, 1 week ago

Selected Answer: BC

B and C

upvoted 1 times

👤 **mattuyghur** 5 months, 1 week ago

Selected Answer: BC

should be B and C

upvoted 1 times

 **Tyler2023** 8 months, 2 weeks ago

I don't think B is an answer here
When you change any configuration manually outside your terraform configuration
The next time you run terraform apply, terraform will revert your manual changes
and follow what is defined in your configuration files

So answer i C

upvoted 1 times

 **brax404** 9 months ago

Selected Answer: BC

B. This is true, but with an important distinction. Terraform doesn't automatically detect changes made outside of it. If you modify resources directly in Azure (outside of Terraform), during the next terraform plan or terraform apply, there might be a discrepancy between the state file and the actual infrastructure. Terraform will then propose to reconcile this difference, which could involve changing or recreating the affected resources.

C. This is true. The Terraform state file isn't automatically updated when changes are made outside of Terraform, such as directly in the Azure Cloud Console.

upvoted 2 times

 **Pikopo** 9 months, 2 weeks ago

the correct answer are B and C

upvoted 1 times

 **gofavad926** 9 months, 2 weeks ago

Selected Answer: BC

BC

When a change is made to the resources via the Azure Cloud Console, the current state file will not be updated.

Terraform will update the state file to reflect the changes made to the resources via the Azure Cloud Console during the next plan or apply.

upvoted 1 times

 **debabrata6983** 10 months, 3 weeks ago

Selected Answer: BC

The correct answer is B&C

upvoted 1 times

 **Rajmane** 11 months, 1 week ago

Selected Answer: BC

The right answer is B & C

upvoted 1 times

 **modarov** 11 months, 2 weeks ago

B. When a change is made to the resources via the Azure Cloud Console, Terraform will update the state file to reflect them during the next plan or apply

C. When a change is made to the resources via the Azure Cloud Console, the current state file will not be updated

upvoted 1 times

 **March2023** 1 year, 1 month ago

Selected Answer: BC

b and c

upvoted 1 times

 **milan92stankovic** 1 year, 1 month ago

Selected Answer: BC

B & C are the correct answer.

upvoted 1 times

 **Taalai** 1 year, 1 month ago

according to Chat GPT is B and D

upvoted 1 times

 **sdm13168** 1 year, 1 month ago

Selected Answer: BC

BC for sure

You need to deploy resources into two different cloud regions in the same Terraform configuration. To do that, you declare multiple provider configurations as follows:

```
provider "aws" {
  region = "us-east-1"
}

provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

What meta-argument do you need to configure in a resource block to deploy the resource to the `us-west-2` AWS region?

- A. alias = west
- B. provider = west
- C. provider = aws.west
- D. alias = aws.west

Correct Answer: C

Reference:

<https://github.com/hashicorp/terraform/issues/451>

Community vote distribution

C (100%)

✉️  **bp39** Highly Voted 2 years, 1 month ago

Selected Answer: C

<https://www.terraform.io/language/providers/configuration>
upvoted 12 times

✉️  **Bere** Most Recent 8 months, 2 weeks ago

Selected Answer: C

For us-east-1:
resource "aws_instance" "example_east" {
ami = "ami-abc12345"
instance_type = "t2.micro"

// No provider meta-argument specified, so it uses the default (us-east-1)
}

For us-west-2:
resource "aws_instance" "example_west" {
ami = "ami-xyz67890"
instance_type = "t2.micro"

provider = aws.west // This directs Terraform to use the 'us-west-2' region provider configuration
}

With this setup, Terraform will deploy the example_east instance in the us-east-1 region and the example_west instance in the us-west-2 region.
upvoted 2 times

✉️  **Ni33** 1 year, 2 months ago

Selected Answer: C

CCCCCCCCCC
upvoted 2 times

✉️  **Power123** 1 year, 3 months ago

Answer is C
upvoted 1 times

✉  **thor7** 1 year, 3 months ago

Selected Answer: C

Definitely, C is the correct answer.

<https://www.terraform.io/language/providers/configuration>

upvoted 2 times

✉  **Nzudin** 1 year, 4 months ago

<https://developer.hashicorp.com/terraform/language/providers/configuration#selecting-alternate-provider-configurations>

upvoted 1 times

✉  **Eltooth** 2 years ago

Selected Answer: C

C is correct answer.

upvoted 1 times

✉  **Zam88** 2 years ago

alias is not meta-argument

c is correct

upvoted 2 times

✉  **elvancedonzy** 2 years, 1 month ago

Selected Answer: C

C is Correct

upvoted 2 times

✉  **concepts015** 2 years, 1 month ago

I think the right answer is D alias is used in the argument and AWS.WEST is the right way to write the argument

upvoted 3 times

✉  **CHRIS12722222** 2 years ago

No, C is correct

upvoted 2 times

✉  **Roro_Brother** 2 years, 1 month ago

Selected Answer: C

C is correct

upvoted 3 times

✉  **bigboi23** 2 years, 1 month ago

Selected Answer: C

Confirmed. C is correct.

upvoted 2 times

✉  **Orpheus123** 2 years, 2 months ago

C - is the correct answer

upvoted 4 times

✉  **traceme** 2 years, 2 months ago

answer is C

upvoted 1 times

- A. Add node_count = var.node_count
- B. Declare the variable in a terraform.tfvars file
- C. Declare a node_count input variable for child module
- D. Nothing, child modules inherit variables of parent module

Correct Answer: C

Community vote distribution

C (82%)

Other

✉  **tbhttp** Highly Voted 1 year, 11 months ago

Selected Answer: C

C is correct as the question is about passing a variable. So if you want the parent to pass its output to its child you need to configure an input variable for that.
The name of the input variable makes no sense but maybe that is to emphasize on the fact that the name of the input variable is arbitrary.

"That module may call other modules and connect them together by passing output values from one to input values of another."
<https://www.terraform.io/language/modules/develop>

upvoted 21 times

✉  **Tyler2023** Highly Voted 8 months, 2 weeks ago

All given options are wrong, they don't make any sense
You have declared a variable called "environment"

A. where did you get the var.node_count, is it related to the question?

B. You can do that, but it is not what you need for the question

C. Then what? node_count variable should accept number, not a string. Assuming "environment" variable will hold string value like Dev, UAT, Prod

D. Child module will not automatically inherit variables from parent module

They don't make any sense

upvoted 9 times

✉  **7b5b962** Most Recent 3 weeks, 5 days ago

```
module "child_module" {
  source = "./path_to_child_module"
  environment = var.environment
}
```

Therefore, the correct answer is:

A. Add node_count = var.node_count

upvoted 1 times

✉  **TigerInTheCloud** 6 months, 4 weeks ago

Selected Answer: A

You need to pass the value.

(The child module should have the variable, that the child module developer's work)

upvoted 1 times

✉  **arunrkaushik** 11 months, 1 week ago

Mr. Nunyabusines, may you like to rewrite the answer as:

```
module "child_module" {
  source = "./parent_module"
  environment = var.environment
}
```

upvoted 3 times

✉  **modarov** 11 months, 2 weeks ago

C. Declare a node_count input variable for child module.

When you declare an input variable in a parent module, the child module will not automatically inherit the value of the variable. You must explicitly declare the variable in the child module.

upvoted 3 times

milan92stankovic 1 year, 1 month ago

Selected Answer: C

I vote for C

upvoted 1 times

Nunyabiznes 1 year, 3 months ago

Selected Answer: C

```
module "child_module" {  
source = "./child_module"
```

```
environment = var.environment  
}
```

upvoted 4 times

camps 1 year, 3 months ago

Selected Answer: C

C. Declare a node_count input variable for the child module.

In Terraform, input variables are used to parameterize Terraform configurations. When using child modules, it is necessary to pass values for input variables defined in the child module. The values for the input variables can be passed using different methods such as using default values, command-line flags, environment variables, variable files, and so on.

upvoted 1 times

thor7 1 year, 3 months ago

Selected Answer: C

<https://www.terraform.io/language/modules/develop>

C is a correct answer

upvoted 1 times

pyro7 1 year, 5 months ago

C. Declare a node_count input variable for child module.

When passing variables from a parent module to a child module in Terraform, you need to explicitly declare the variables in both the parent and the child modules. In this case, you would need to declare an input variable for the child module that corresponds to the input variable declared in the parent module. So, the correct option is C.

Option A is incorrect because it is using an example variable that is not related to the input variable mentioned in the question. Option B is incorrect because it refers to how to set the value of the variable, but not how to pass the value from the parent to the child module. Option D is also incorrect because child modules do not automatically inherit the variables of their parent modules.

upvoted 3 times

gekkehenk 1 year, 6 months ago

Selected Answer: C

C, declaration of variables is not done in terraform.tfvars.

upvoted 2 times

DerekKey 1 year, 7 months ago

C - correct

for every variable used in terraform module (root/child) you must first declare it.

Without declaration in child module the variable will not be recognized.

upvoted 2 times

robertninho 1 year, 6 months ago

```
# Parent module
```

```
module "child" {  
source = "./child"  
node_count = var.environment  
}
```

```
# Child module
```

```
variable "node_count" {  
type = number  
}
```

upvoted 7 times

 **yogishrb2020** 1 year, 9 months ago

Selected Answer: C

Declare the variable in a terraform.tfvars file --> doesnot apply to child module
--><https://www.terraform.io/language/values/variables>
upvoted 2 times

 **nani1709** 1 year, 9 months ago

A is the correct answer.
upvoted 1 times

 **Linus11** 1 year, 9 months ago

Searched in google " terraform node_count"
no result.
What is this node_count do?
upvoted 2 times

 **donathon** 1 year, 11 months ago

Selected Answer: B

The answer is B. TFCVARS is like a global variable.
upvoted 2 times

 **tbhttp** 1 year, 11 months ago

B is not correct. Reading something from the same place is not passing a variable.
Imagine what happens if the parent gets its value passed with -var.
upvoted 2 times

Question #101

Topic 1

If a module declares a variable with a default, that variable must also be defined within the module.

A. True

B. False

Correct Answer: A

Community vote distribution

B (100%)

 **govs**  1 year, 3 months ago

Selected Answer: B

If a module declares a variable with a default value, it is not necessary to define the variable within the module calling the module. The module will automatically use the default value if a value is not explicitly assigned to the variable in the calling module
upvoted 8 times

 **Eltooth**  2 years ago

Selected Answer: B

B is correct answer.
upvoted 7 times

Clapton79 Most Recent 8 months, 1 week ago

This question does not make sense in its current form. (Very poor English, sorry)

You declare the variable with variable "my_var" {type=string} and in there you may add a default: variable "my_var" {
type=string
default="a"
}

SETTING a variable in a statement calling a module that contains this variable is not compulsory.

```
module "my_module" {  
source= "./this_module"  
my_var = "b" # <-- you can omit this in which case the value will be "a"  
}
```

upvoted 5 times

umavaja 7 months, 1 week ago

Agree, very poor English, same issue even for UDEMY courses as well Terraform documentation

upvoted 3 times

otakuinside 9 months, 2 weeks ago

The variable must be declared aswell, another thing is that you have to assign a value to it. But the declaration is mandatory.
At least this is my understanding of what the question says.

upvoted 1 times

Alex1778 1 year, 3 months ago

B is correct answer.

upvoted 1 times

Power123 1 year, 3 months ago

Bis correct

upvoted 1 times

Nunyabiznes 1 year, 3 months ago

Selected Answer: B

False.

If a module declares a variable with a default value, it does not necessarily need to be defined within the module. The default value will be used if the variable is not set in the calling module.

For example, consider the following module:

```
variable "instance_type" {  
description = "The instance type to use"  
default = "t2.micro"  
}
```

```
resource "aws_instance" "example" {  
ami = "ami-0c55b159cbfafe1f0"  
instance_type = var.instance_type  
}
```

upvoted 4 times

Nunyabiznes 1 year, 3 months ago

In this module, a variable called instance_type is declared with a default value of t2.micro. The default value will be used if the variable is not set in the calling module.

If the calling module does not set the instance_type variable, the default value of t2.micro will be used. If the calling module does set the instance_type variable, the value from the calling module will be used instead of the default.

It is important to note that if a variable is declared without a default value and is not set in the calling module, Terraform will prompt the user to enter a value for that variable.

upvoted 6 times

SilentMilli 1 year, 4 months ago

Selected Answer: B

A module can declare a variable with a default value without defining it within the module. This is because the variable can be used as-is with a default value, or it can be assigned a new value by code that imports the module.

upvoted 2 times

✉  **bukake** 1 year, 11 months ago

Where are 102-103-104 Questions?

upvoted 4 times

✉  **kopper2019** 1 year, 10 months ago

no idea, I am taking exam tomorrow
upvoted 2 times

✉  **hello2022** 1 year, 10 months ago

How did you do? How relevant were these questions.

upvoted 1 times

✉  **gsx** 2 years, 1 month ago

Selected Answer: B

when no value is passed, default value will be automatically be taken.

upvoted 4 times

✉  **Maaran07** 2 years, 1 month ago

It is not mandatory to define inside module. default can be in module itself.

upvoted 1 times

✉  **traceme** 2 years, 2 months ago

Why not A . When we define the variable as default , we need to pass tha value for the variables right

upvoted 2 times

✉  **habros** 2 years, 1 month ago

It already had a variable. Worse case it knows where to get if it's not manually defined

upvoted 1 times

Question #102

Topic 1

Which option cannot be used to keep secrets out of Terraform configuration files?

- A. Environment Variables
- B. Mark the variable as sensitive
- C. A Terraform provider
- D. A -var flag

Correct Answer: D

Community vote distribution

B (60%)

C (30%)

10%

✉  **rotimislaw**  1 year, 4 months ago

Selected Answer: B

It's B

The sensitive flag only prevents secret from showing up in the CLI outputs, but the value itself is still present in the configuration files.
upvoted 17 times

✉️  **Chrisler** 10 months, 1 week ago

This is true, you can still use the answer D. A terraform provider (e.g. Vault) to keep the secrets out of the terraform configuration.
upvoted 2 times

✉️  **MauroSoli** 8 months, 3 weeks ago

The question is "Which option CANNOT be used"
upvoted 5 times

✉️  **Tyler2023** 8 months, 2 weeks ago

I accidentally click the like button :D
I agree with rotimislaw, the answer is B
upvoted 1 times

✉️  **zaaath** Highly Voted  1 year, 3 months ago

Selected Answer: B

It's B. I think ChatGPT gets confused with this question, and people end up posting its response here.

A. Environment Variables

- keeps the value out of configuration and state

B. Mark the variable as sensitive

- the value is still in the configuration and state, but not in the console output

C. A Terraform provider

- keeps the value out of configuration, but not the state (like a provider's data resource)

D. A -var flag

- keeps the value out of configuration, but not the state

upvoted 8 times

✉️  **zaaath** 1 year, 3 months ago

Correction:

D. A -var flag

- keeps the value out of configuration and state

upvoted 2 times

✉️  **a54b16f** Most Recent  5 months, 1 week ago

Selected Answer: B

C refers to Vault, which is valid

upvoted 1 times

✉️  **vipulchoubisa** 6 months, 1 week ago

D--VAR OPTION IS OUTSIDE .TF FILE.

A-CAN BE SET IF YOU ARE USING TERRAFORM CLOUD

B-CAN BE SET UNDER VARIABLE.TF FILE

C-CAN BE SET UNDER PROVIDER BLOCK WHICH IS ALSO ONE OF THE .TF FILE

ONLY -VAR FLAG IS OUTSIDE TERRAFROM, HENCE THIS IS CORRECT.

upvoted 1 times

✉️  **jutove_mi** 6 months, 1 week ago

Using Option B we can keep secrets out of Terraform configuration files, but question asked which option "can not be used" ? so I suppose C (also chatgpt reply C)

upvoted 1 times

✉️  **Arshad011294** 7 months ago

Selected Answer: B

I will go for B, for anyone thinking its C, there is a video in this official terraform documentation where a lady is explaining best practices of using provider vault, where we can set secrets to expire and hence the actual secrets are kept from getting revealed in the terraform state file.
<https://registry.terraform.io/providers/hashicorp/vault/latest/docs>

upvoted 2 times

👤 **Bere** 7 months, 3 weeks ago

Selected Answer: B

Answer is B. Mark the variable as sensitive.

- A. Environment Variables: This method keeps secrets out of configuration files by setting them externally.
- B. Mark the variable as sensitive: While this marks the variable as sensitive in Terraform's state and prevents it from being displayed in CLI outputs, the secret can still be present in the configuration files and state file.
- C. A Terraform provider: This doesn't inherently store or hide secrets in configuration files.
- D. A -var flag: This method can be used to pass variables at runtime, thus keeping them out of configuration files.

upvoted 2 times

👤 **MisterROBOT** 8 months, 2 weeks ago

B

Terraform documentation

When you mark a variable as sensitive in Terraform, it does not keep the secrets out of the configuration files . Instead, it makes the variable write-only and prevents all users from viewing its value in the Terraform Cloud UI or reading it through the Variables API endpoint . Users with permission to read and write variables can set new values for sensitive variables, but you must delete and recreate the variable to edit its other attributes.

upvoted 1 times

👤 **brax404** 9 months ago

Selected Answer: C

C. A Terraform provider: Terraform providers are plugins that are responsible for managing the lifecycle of resources. They don't inherently offer a mechanism to keep secrets out of Terraform configuration files.

upvoted 4 times

👤 **debabrata6983** 10 months, 3 weeks ago

Selected Answer: B

Secret flag prevents showing up the value in CLI o/p

upvoted 1 times

👤 **BaburTurk** 10 months, 3 weeks ago

Selected Answer: C

C. A Terraform provider.

Terraform providers are used to interact with specific cloud providers or other APIs. They do not have the ability to access or store secrets.

The other options, environment variables, marking the variable as sensitive, and the -var flag, can all be used to keep secrets out of Terraform configuration files.

Environment variables are a common way to pass secrets to Terraform. They are stored outside of the Terraform configuration files, so they are not visible to anyone who can read the Terraform code.

Marking the variable as sensitive tells Terraform to encrypt the variable value when it is stored in the state file. This helps to protect the secret value from being exposed.

The -var flag can be used to pass secret values to Terraform from the command line. This is useful for one-off operations, but it is not a recommended way to manage secrets.

upvoted 2 times

👤 **Tyler2023** 8 months, 2 weeks ago

I think Terraform will not encrypt your secrets in your state file, unless you are using Terraform cloud or enterprise

upvoted 1 times

✉️  **kudakk** 11 months ago

Selected Answer: D

GPT:

D. A -var flag

Using the -var flag when running terraform apply or terraform plan from the command line requires you to specify variable values directly in the command, which could expose sensitive information in the command line history, system process list, or in logs.

While the -var flag is a way to set variables in Terraform, it's not a recommended way to handle sensitive information due to the exposure risk associated with command line arguments.

upvoted 1 times

✉️  **Chrisler** 10 months, 1 week ago

But based on the question, "secrets out of Terraform configuration files" Yes it is in the CLI history, sys process list or in logs, but not the terraform configuration.

upvoted 1 times

✉️  **arunrkaushik** 11 months, 1 week ago

Try to understand the question, it is important :

What is one method that cannot be utilized to prevent secrets from appearing in Terraform configuration files?

What approach does not work to exclude confidential data from being included in Terraform config files?

Which technique is not valid for keeping sensitive information out of the configs managed by Terraform?

What is an invalid solution for ensuring private credentials do not get embedded in a Terraform configuration?

Which choice is not viable for stopping secret keys and access tokens from being added to Terraform's setup files?

What method fails to protect classified details from being inserted into the configuration code used by Terraform?

upvoted 1 times

✉️  **milan92stankovic** 1 year, 1 month ago

Selected Answer: B

I vote for B

upvoted 1 times

✉️  **sdm13168** 1 year, 2 months ago

Selected Answer: B

tested

upvoted 1 times

✉️  **kiran15789** 1 year, 2 months ago

Selected Answer: D

A -var flag

upvoted 2 times

✉️  **Rajsp** 1 year, 3 months ago

Answer is C.

A. Environment variables: You can use environment variables to provide values for variables used in your Terraform configuration files. This way sensitive data can be stored outside the configuration files and not committed to version control.

B. Mark the variable as sensitive: You can mark a variable as sensitive in your Terraform configuration file by setting the sensitive argument to true. This will prevent the variable from being displayed in the output of the terraform plan and terraform apply commands.

D. A -var flag: You can provide variable values at the command line using the -var flag when running the terraform plan and terraform apply commands. This way, you can pass sensitive data to Terraform without having to store it in configuration files.

upvoted 4 times

Question #103

Topic 1

Which of the following arguments are required when declaring a Terraform output?

A. sensitive

B. description

C. default

D. value

Correct Answer: D

Community vote distribution

D (100%)

 **Nunyabiznes** Highly Voted 1 year, 3 months ago

Selected Answer: D

The only required argument when declaring a Terraform output is the value argument, which specifies the value of the output:

```
```  
output "example" {
 value = "example output value"
}
```
```

The description argument is optional and can be used to provide additional information about the output:

```
```  
output "example" {
 value = "example output value"
 description = "An example output"
}
```
```

upvoted 9 times

 **Nunyabiznes** 1 year, 3 months ago

The sensitive argument is also optional and can be used to mark the output as sensitive, which prevents its value from being displayed in the output of Terraform commands, logs, and state files:

```
```  
output "example" {
 value = "example output value"
 sensitive = true
}
```
```

The default argument is not a valid argument for an output. It is used for input variables and specifies a default value to be used if a value is not set in the configuration or passed in through the command line.

upvoted 3 times

 **Ni33** Most Recent 1 year, 2 months ago

Selected Answer: D

DDDDDDDDDD

upvoted 1 times

 **Power123** 1 year, 3 months ago

D is correct

upvoted 1 times

 **nakikoo** 1 year, 6 months ago

Selected Answer: D

<https://developer.hashicorp.com/terraform/language/values/outputs>

upvoted 3 times

 **Gaby999** 1 year, 10 months ago

yes, correct answer is D

upvoted 1 times

 **Burakko** 1 year, 10 months ago

Selected Answer: D

There has to be a value for sure.

upvoted 2 times

Your risk management organization requires that new AWS S3 buckets must be private and encrypted at rest. How can Terraform Enterprise automatically and proactively enforce this security control?

- A. With a Sentinel policy, which runs before every apply
- B. By adding variables to each TFE workspace to ensure these settings are always enabled
- C. With an S3 module with proper settings for buckets
- D. Auditing cloud storage buckets with a vulnerability scanning tool

Correct Answer: B

Community vote distribution

A (100%)

camps Highly Voted 1 year, 3 months ago

Selected Answer: A

- A. With a Sentinel policy, which runs before every apply.

Terraform Enterprise can enforce security controls through the use of Sentinel policies. Sentinel is a policy as code framework that integrates with Terraform Enterprise and can be used to enforce specific security controls. In this case, the Sentinel policy could check that all new S3 buckets are set to be private and encrypted at rest and prevent the Terraform apply from proceeding if the buckets do not meet this requirement. This ensures that the security control is automatically and proactively enforced every time Terraform makes changes to the infrastructure.

upvoted 9 times

selvaraj133ece Most Recent 11 months, 2 weeks ago

Answer, B only. They want to keep the S3 bucket private. So, it will be a different state file.

upvoted 1 times

Rohit000003 1 year ago

Selected Answer: A

As per terraform document

upvoted 1 times

Ni33 1 year, 2 months ago

Selected Answer: A

AAAAAAAAAA

upvoted 1 times

Power123 1 year, 3 months ago

A is correct

upvoted 1 times

Nunyabiznes 1 year, 3 months ago

Selected Answer: A

import "tfplan"

```
# Ensure all new S3 buckets are private and encrypted at rest
deny[msg] {
  resources := tfplan.module_paths["aws_s3_bucket"]
  not all_true([
    for r in resources:
      r.attributes.acl == "private" and
      r.attributes.server_side_encryption_configuration.0.rule.0.apply_server_side_encryption_by_default.0.sse_algorithm == "AES256"
  ])
  msg := "All new S3 buckets must be private and encrypted at rest"
}
```

upvoted 3 times

 **SilentMilli** 1 year, 4 months ago

Selected Answer: A

Terraform Enterprise provides the ability to enforce security controls through Sentinel policies, which are a form of policy as code. Sentinel policies allow you to define and enforce organizational or regulatory policies by creating a set of rules that run before each Terraform operation.

upvoted 2 times

 **Ame2222** 1 year, 5 months ago

A is correct

upvoted 1 times

 **Daro_** 1 year, 5 months ago

Selected Answer: A

yes A Corrcrt

upvoted 1 times

 **seif1993** 1 year, 7 months ago

yes A Corrcrt

upvoted 1 times

 **RVivek** 1 year, 9 months ago

Selected Answer: A

Sentinel policy is the best way to manage multiple workspaces

upvoted 1 times

 **bora4motion** 1 year, 10 months ago

Selected Answer: A

I go with A.

upvoted 2 times

 **Burakko** 1 year, 10 months ago

Selected Answer: A

With a Sentinel policy for sure.

upvoted 2 times

 **mav3r1ck** 1 year, 10 months ago

A.

Reference:

<https://docs.hashicorp.com/sentinel/intro/what>

<https://medium.com/hashicorp-engineering/enforcing-aws-s3-security-best-practice-using-terraform-sentinel-ddcd181ff4b7>

upvoted 4 times

Most Terraform providers interact with _____.

- A. API
- B. VCS Systems
- C. Shell scripts
- D. None of the above

Correct Answer: A

Community vote distribution

A (100%)

 **Hizumi** Highly Voted 1 year, 10 months ago

Answer is A.

Terraform relies on plugins called "providers" to interact with cloud providers, SaaS providers, and other APIs, as per:
<https://www.terraform.io/language/providers>

upvoted 5 times

 **Boombay_420** Most Recent 8 months, 3 weeks ago

Selected Answer: A

A is the right answer

upvoted 1 times

 **debabrata6983** 10 months, 3 weeks ago

Selected Answer: A

Provider plugin abstracts the API interaction with the cloud provider.

upvoted 3 times

 **SairamObili** 11 months ago

it's API

Answer is A

upvoted 1 times

 **IK912** 1 year ago

A is the answer

upvoted 1 times

 **Power123** 1 year, 3 months ago

A is correct

upvoted 1 times

 **seif1993** 1 year, 7 months ago

Selected Answer: A

yes for A

upvoted 1 times

 **donathon** 1 year, 10 months ago

Selected Answer: A

A is the answer

upvoted 1 times

QUESTION Validate variables that your infrastructure matches the Terraform state file.

A. True

B. False

Correct Answer: A

Community vote distribution

B (100%)

 **RVivek** Highly Voted 1 year, 9 months ago

Selected Answer: B

Validate only checks the syntax

<http://man.hubwiz.com/docset/Terraform.docset/Contents/Resources/Documents/docs/commands/validate.html>

upvoted 10 times

 **babgad** Most Recent 10 months, 3 weeks ago

Selected Answer: B

False B

Validate runs checks that verify whether a configuration is syntactically valid and internally consistent, regardless of any provided variables or existing state. It is thus primarily useful for general verification of reusable modules, including correctness of attribute names and value types.

It is safe to run this command automatically, for example as a post-save check in a text editor or as a test step for a re-usable module in a CI system

<https://developer.hashicorp.com/terraform/cli/v1.1.x/commands/validate>

upvoted 2 times

 **Busi57** 11 months, 4 weeks ago

Selected Answer: B

false B Validate only checks the syntax

upvoted 2 times

 **Busi57** 11 months, 4 weeks ago

False B

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

B is the correct answer. Terraform plan validates desired state with current state configuration.

upvoted 2 times

 **Power123** 1 year, 3 months ago

False , B is correct

upvoted 1 times

 **adouban** 1 year, 7 months ago

Selected Answer: B

B is correct

upvoted 1 times

 **Optimus** 1 year, 10 months ago

Selected Answer: B

B because Terraform validate only checks for syntax error or so

upvoted 2 times

 **bora4motion** 1 year, 10 months ago

Selected Answer: B

It's more of a syntax thing.

upvoted 2 times

 **Burakko** 1 year, 10 months ago

Selected Answer: B

False, terraform validate has nothing to do with the state file.

upvoted 2 times

 **Uma10** 1 year, 10 months ago

Selected Answer: B

The terraform validate command validates the configuration files in a directory, referring only to the configuration and not accessing any remote services such as remote state, provider APIs, etc.

Validate runs checks that verify whether a configuration is syntactically valid and internally consistent, regardless of any provided variables or existing state. It is thus primarily useful for general verification of reusable modules, including correctness of attribute names and value types.

Source: <https://www.terraform.io/cli/commands/validate>

upvoted 3 times

Question #107

Topic 1

What does terraform import allow you to do?

- A. Import a new Terraform module
- B. Use a state file to import infrastructure to the cloud
- C. Import provisioned infrastructure to your state file
- D. Import an existing state file to a new Terraform workspace

Correct Answer: C

Community vote distribution

C (100%)

 **Ni33** 1 year, 2 months ago

Selected Answer: C

Yes, C is the correct answer.

upvoted 2 times

 **Power123** 1 year, 3 months ago

C is correct

upvoted 1 times

 **bora4motion** 1 year, 10 months ago

Selected Answer: C

I go with C

upvoted 4 times

 **Burakko** 1 year, 10 months ago

Selected Answer: C

%100 true

upvoted 2 times

Question #108

Topic 1

FILL BLANK -

In the below configuration, how would you reference the module output vpc_id?

```
module "vpc" {  
    source = "terraform-and-modules/vpc/aws"
```

```
source = "community-modules/vpc/aws"
cidr = "10.0.0.0/16"
name = "test-vpc"
}
```

Type your answer in the field provided. The text field is not case sensitive and all variations of the correct answer are accepted.

Correct Answer: *output "outvpc_id"*

✉️  **Hizumi** Highly Voted 1 year, 10 months ago

module.vpc.vpc_id
Reference: <https://cloudcasts.io/course/terraform/community-vpc-module>
upvoted 70 times

✉️  **shopkitty** Highly Voted 1 year, 10 months ago

it should be module.vpc.vpc_id
upvoted 18 times

✉️  **temor** Most Recent 3 months, 2 weeks ago

module.vpc.vpc_id
upvoted 2 times

✉️  **5719d28** 4 months, 2 weeks ago

module.vpc.vpc_id
upvoted 2 times

✉️  **NashP** 5 months, 2 weeks ago

To reference the module output `vpc_id` in the given configuration, you would use the following syntax:

```
```hcl
module.vpc.vpc_id
````
```

This assumes that the output variable `vpc_id` is declared in the "example" module. Replace `vpc` with the actual name you gave to your module. So, if your module is named "vpc," the reference would be:

```
```hcl
module.vpc.vpc_id
````
```

This syntax allows you to access the output value `vpc_id` from the specified module in your Terraform configuration.
upvoted 1 times

✉️  **KingFvsher** 10 months ago

You can reference the vpc_id output of a module in Terraform using the module.<MODULE_NAME>.<OUTPUT_NAME> syntax. In this case, if the name of the module block is vpc, you would reference the vpc_id output as module.vpc.vpc_id.
upvoted 5 times

✉️  **debabrata6983** 10 months, 3 weeks ago

module.vpc.vpc_id
upvoted 2 times

✉️  **arunrkaushik** 11 months, 1 week ago

```
resource "aws_vpc" "my_vpc" {
  cidr_block = "10.0.0.0/16"
  ...
}

output "vpc_id" {
  value = aws_vpc.my_vpc.id
}
````
```

upvoted 1 times

✉️  **joyboy23** 1 year ago

module.vpc.vpc\_id  
upvoted 1 times

abhi6199 1 year ago

its module.vpc.vpc\_id

upvoted 1 times

IK912 1 year ago

module.vpc.vpc\_id

upvoted 1 times

Ni33 1 year, 2 months ago

module.vpc.vpc\_id

upvoted 1 times

Nunyabiznes 1 year, 3 months ago

To reference the vpc\_id output of the vpc module in the above configuration, you would use the following syntax:

module.vpc.vpc\_id

This would allow you to reference the vpc\_id output from other parts of your Terraform configuration, such as when creating resources that depend on the VPC. For example:

```
resource "aws_subnet" "subnet_1" {
 vpc_id = module.vpc.vpc_id
 cidr_block = "10.0.1.0/24"
}
```

In this example, the aws\_subnet resource is referencing the vpc\_id output of the vpc module to ensure that the subnet is created in the correct VPC.

upvoted 5 times

alexandroe 1 year, 4 months ago

module.vpc.id

upvoted 1 times

TechHero 1 year, 5 months ago

To reference the output vpc\_id from the above module configuration, you would use the following syntax:

module.vpc.vpc\_id

Here, module is the prefix used to reference outputs from a module, vpc is the name of the module, and vpc\_id is the name of the output being referenced.

upvoted 5 times

alec123 1 year, 5 months ago

correct

upvoted 1 times

RVivek 1 year, 9 months ago

It should be module.vpc.vpc\_id

upvoted 2 times

keiffo2 1 year, 10 months ago

module.vpc.id

upvoted 4 times

How would you reference the Volume IDs associated with the ebs\_block\_device blocks in this configuration?

```
resource "aws_instance" "example" {
 ami = "ami-abc123"
 instance_type = "t2.micro"

 ebs_block_device {
 device_name = "sda2"
 volume_size = 16
 }

 ebs_block_device {
 device_name = "sda3"
 volume_size = 20
 }
}
```

- A. aws\_instance.example.ebs\_block\_device.[\*].volume\_id
- B. aws\_instance.example.ebs\_block\_device.volume\_id
- C. aws\_instance.example.ebs\_block\_device[sda2,sda3].volume\_id
- D. aws\_instance.example.ebs\_block\_device.\*.volume\_id

**Correct Answer: C**

*Community vote distribution*

D (84%)

A (16%)

 **KJ\_Rollings** Highly Voted 1 year, 1 month ago

**Selected Answer: D**

It's D.

If you're using square brackets, then it should be like,  
ebs\_block\_device[\*].volume\_id

upvoted 17 times

 **fedex** Highly Voted 1 year, 6 months ago

100% D.

aws\_instance.example.ebs\_block\_device.\*.volume\_id and aws\_instance.example.ebs\_block\_device[\*].volume\_id are equivalent and will both refer to the volume\_id field for all of the EBS block devices that are attached to the aws\_instance resource with the name example.

The \* syntax is known as the "splat" operator and can be used to indicate that the preceding attribute should be repeated for each element in list. It can be used with both the . and [] notation.

For example, suppose you have an aws\_instance resource named example that has two EBS block devices attached to it, with volume\_id val of vol-12345678 and vol-87654321. You could use the \* operator like this:

```
aws_instance.example.ebs_block_device.*.volume_id
This will evaluate to ["vol-12345678", "vol-87654321"]
```

```
aws_instance.example.ebs_block_device[*].volume_id
This will also evaluate to ["vol-12345678", "vol-87654321"]
```

In both cases, the volume\_id field is being accessed for each element in the ebs\_block\_device list, and the resulting value is a list of the volume\_id values for each element.

upvoted 17 times

✉️  **kennywuu** Most Recent 6 months, 2 weeks ago

it is D.

Just check it in terraform console, only D is passed :)

upvoted 4 times

✉️  **TigerInTheCloud** 6 months, 4 weeks ago

Selected Answer: D

A is almost right and better and more modern if there is no extra dot

You can add an output and validate each of the expressions:

```
output "ebs-volume-ids" {
 value = aws_instance.example.ebs_block_device["sda2"].volume_id
}
```

upvoted 1 times

✉️  **TigerInTheCloud** 6 months, 4 weeks ago

C can rewrite as:

```
[for v in ["sda2", "sda3"] : aws_instance.example.ebs_block_device[v].volume_id]
```

upvoted 1 times

✉️  **Tyler2023** 8 months, 2 weeks ago

All of them are invalid expression

Based on this <https://developer.hashicorp.com/terraform/language/expressions/references>

The arguments of the ebs\_block\_device nested blocks can be accessed using a splat expression. For example, to obtain a list of all of the device\_name values, use aws\_instance.example.ebs\_block\_device[\*].device\_name.

upvoted 3 times

✉️  **MauroSoli** 8 months, 3 weeks ago

The answer is D but it's a form deprecated

upvoted 1 times

✉️  **amehim** 9 months, 1 week ago

```
I with D: > aws_instance.app_server.ebs_block_device.*.volume_id
tolist([
 "vol-0aa30805cf71780e4",
 "vol-081f89b26f2fed5ac",
])
```

upvoted 1 times

✉️  **vj\_dhaksh** 1 year, 1 month ago

answer is D.. <<<Tested >>>

upvoted 4 times

✉️  **Nunyabiznes** 1 year, 3 months ago

Selected Answer: D

D. aws\_instance.example.ebs\_block\_device.\*.volume\_id

To reference the Volume IDs associated with the ebs\_block\_device blocks in the given aws\_instance resource, you can use the .\* syntax to reference all elements of the list. The ebs\_block\_device blocks are represented as a list in the aws\_instance resource, and each block has a volume\_id attribute that you can reference.

upvoted 4 times

✉️  **lordoftheringsnewavatar** 1 year, 3 months ago

```
output "splat" {
 value = aws_instance.test-vm.ebs_block_device.*.volume_id
}
```

Outputs:

```
splat = tolist([
 "vol-088ecd़fa25d46bb49",
 "vol-030bfe9d14a06ecd2",
])
```

Selected Answer: D

upvoted 1 times

✉  **alexandroe** 1 year, 4 months ago

**Selected Answer: D**

one valid expierience

```
resource "aws_instance" "example" {
 ami = "ami-a10b4dc2"
 instance_type = "t2.micro"

 # Connect the instance to the EBS Volumes attached
 # https://www.terraform.io/docs/providers/aws/r/instance.html#ebs_block_device
 ebs_block_device {
 device_name = "/dev/sda1"
 volume_size = 10
 volume_type = "gp2"
 }
 ebs_block_device {
 device_name = "/dev/sdb1"
 volume_size = 20
 volume_type = "gp2"
 }
}

output "splat" {
 value = ["${aws_instance.example.*.id}", "${aws_ebs_volume.example.*.id}"]
}
```

upvoted 2 times

✉  **awsguys** 1 year, 5 months ago

D => Right

upvoted 2 times

✉  **gekkehenk** 1 year, 6 months ago

**Selected Answer: D**

The brackets do not work, asterisk only.

upvoted 4 times

✉  **resnef** 1 year, 6 months ago

**Selected Answer: D**

Answer is D

```
> aws_instance.example.ebs_block_device.*.volume_id
tolist([
"vol-123457",
"vol-43437834743348",
])

> aws_instance.example.ebs_block_device.[*].volume_id
|
| Error: Invalid attribute name
|
| on <console-input> line 1:
| (source code not available)
|
| An attribute name is required after a dot.
```

upvoted 3 times

✉  **AB7088** 1 year, 6 months ago

If remove the dot then the answer should be A. I think there has a typo.. D is not correct.

upvoted 2 times

✉  **BaburTurk** 1 year, 6 months ago

```
aws_instance.example.ebs_block_device[*].volume_id
or
aws_instance.example.ebs_block_device.*.volume_id
```

then D is the answer

upvoted 2 times

 **secdaddy** 1 year, 7 months ago

```
resource "aws_instance" "example" {
 ami = "ami-0f15e0a4c8d3ee5fe"
 instance_type = "t2.micro"

 ebs_block_device {
 device_name = "/dev/sda1"
 volume_size = "16"
 }
 ebs_block_device {
 device_name = "/dev/sda2"
 volume_size = "20"
 }
}

output "device-list" {
 value = aws_instance.example.ebs_block_device[""].volume_id
}
```

A as written fails. A without the period after device works

```
| 24: value = aws_instance.example.ebs_block_device["].volume_id
| An attribute name is required after a dot.
```

B fails

```
| 24: value = aws_instance.example.ebs_block_device.volume_id
| Block type "ebs_block_device" is represented by a set of objects, and set elements do not have addressable keys. To find elements match
```

C fails (even though there is a closing bracket)

```
| 24: value = aws_instance.example.ebs_block_device[sda2,sda3].volume_id
| The index operator must end with a closing bracket ("").
```

D works as written

upvoted 2 times

What does state locking accomplish?

- A. Copies the state file from memory to disk
- B. Encrypts any credentials stored within the state file
- C. Blocks Terraform commands from modifying the state file
- D. Prevents accidental deletion of the state file

**Correct Answer:** B

*Community vote distribution*

C (100%)

✉  **Uma10** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

If supported by your backend, Terraform will lock your state for all operations that could write state. This prevents others from acquiring the lock and potentially corrupting your state.

Source: <https://www.terraform.io/language/state/locking>  
upvoted 8 times

✉  **chael88** Highly Voted 1 year, 5 months ago

Just passed the exam. There was a similar question not listed here on ExamTopic:

What backend types supports state locking?

- A. local
- B. None
- C. All
- D. Some

I chose D. Some as according to Hashicorp document: "Not all backends support locking"

Source: <https://developer.hashicorp.com/terraform/language/state/backends>

upvoted 6 times

✉  **phidelics** 1 year, 3 months ago

Not all backends supports state locking. you are right with option D

upvoted 2 times

✉  **Ni33** Most Recent 1 year, 2 months ago

**Selected Answer: C**

C is the right answer

upvoted 1 times

✉  **Power123** 1 year, 3 months ago

C is correct

upvoted 1 times

✉  **adouban** 1 year, 7 months ago

**Selected Answer: C**

C state locking is protect state file from modification at the same time

upvoted 2 times

 **barracouto** 1 year, 9 months ago

**Selected Answer: C**

I really wonder if exam topics needs to display the incorrect answers for legality purposes.  
It's C.

upvoted 3 times

 **secdaddy** 1 year, 7 months ago

They're guessing at answers and then refining by crowdsourcing answers from us  
upvoted 1 times

 **asudhin** 1 year, 9 months ago

**Selected Answer: C**

It's C

upvoted 1 times

 **legendary7** 1 year, 9 months ago

C-is the correct answer  
<https://www.terraform.io/language/state/locking>  
upvoted 1 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: C**

Blocks Terraform commands from modifying the state file. Since already one command is doing a modification and mutiple simultaneous writes can make it inconsistant

upvoted 1 times

 **bora4motion** 1 year, 10 months ago

**Selected Answer: C**

I go with C

upvoted 3 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: C**

It just locks the state file not to be modified.

upvoted 4 times

Question #111

Topic 1

You just upgraded the version of a provider in an existing Terraform project. What do you need to do to install the new provider?

- A. Run terraform apply -upgrade
- B. Run terraform init -upgrade
- C. Run terraform refresh
- D. Upgrade your version of Terraform

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Hizumi** Highly Voted 1 year, 10 months ago

Answer is B.

[`-upgrade`] - Opt to upgrade modules and plugins as part of their respective installation steps. See the sections below for more details.  
Reference: <https://www.terraform.io/cli/commands/init#upgrade>

upvoted 16 times

 **Prince\_devops** Most Recent 5 months, 3 weeks ago

Selected Answer: B

Answer is B

upvoted 1 times

 **Tyler2023** 8 months, 2 weeks ago

<https://developer.hashicorp.com/terraform/tutorials/configuration-language/provider-versioning#upgrade-the-aws-provider-version>  
upvoted 1 times

 **IK912** 1 year ago

its B, I tried this

upvoted 1 times

 **Ni33** 1 year, 2 months ago

Selected Answer: B

BBBBBBBB

upvoted 1 times

 **srcntp** 1 year, 6 months ago

Selected Answer: B

Answer is B.

upvoted 1 times

 **hectordj** 1 year, 8 months ago

Selected Answer: B

Answer is B.

upvoted 1 times

 **Bortoloto** 1 year, 9 months ago

Selected Answer: B

Answer is B.

upvoted 1 times

A module can always refer to all variables declared in its parent module.

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

B (66%)

A (34%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: B**

Modules do not inherit variables from the parent module. All modules are self-contained units. So you have to explicitly define variables in the child module, and then explicitly set these variables in the parent module, when you instantiate the child module.

upvoted 25 times

 **Burakko** 1 year, 10 months ago

However you can refer to them from the parent module, I guess it is A. The question is asking for referring not inheriting.

upvoted 13 times

 **kiran15789** Highly Voted 1 year, 2 months ago

**Selected Answer: B**

A module in Terraform can only access variables that are explicitly passed as input variables or declared as outputs by the parent module. It cannot access variables declared within the parent module unless they are passed to the module as input variables. This is known as the "principle of least privilege," which means that modules should only have access to the information and resources that they need to perform their tasks, and no more.

So, to refer to a variable from a parent module, you must either pass it as an input variable to the child module or declare it as an output variable in the parent module and then reference it in the child module using the `module.<name>.<output>` syntax.

upvoted 8 times

 **Tyler2023** 8 months, 2 weeks ago

The question is asking if we can always refer to all variables declared which is true,

It is not asking if it is automatically or manually

upvoted 1 times

 **Tyler2023** 8 months, 2 weeks ago

Answer is A

upvoted 1 times

 **Demonik** Most Recent 7 months, 2 weeks ago

**Selected Answer: A**

A for me

upvoted 3 times

 **Stargazer11** 8 months, 2 weeks ago

Answer: B

A child module can refer to variables declared in its parent module, but it does so in a controlled and explicit manner. By default, child modules do not have automatic access to all variables in the parent module. To reference a variable from the parent module in a child module, you must declare an input variable with the same name in the child module and then explicitly pass the value from the parent module.

upvoted 2 times

 milan92stankovic 1 year, 1 month ago

**Selected Answer: A**

The answer is A.  
You CAN always REFER. You don't have to, but you can.  
upvoted 4 times

 Ni33 1 year, 2 months ago

**Selected Answer: B**

B is the correct answer  
upvoted 1 times

 raf314 1 year, 2 months ago

**Selected Answer: B**

The question is whether a (child) module can always refer to ALL variables in the parent module. The answer is NO. The child module can "see" and refer to the variables defined in its own (child module's) definition. The child module has no visibility of the variables defined in the parent module. Thus it cannot refer to them. Answer is False (B)

upvoted 6 times

 BennaniHaythem 1 year, 3 months ago

A module can refer to variables in its parent module only if those variables are explicitly passed to it as input variables. If a variable is not passed to a child module, then that child module cannot refer to it. So the correct answer is B - False.

upvoted 3 times

 sylvergorilla 1 year, 3 months ago

**Selected Answer: B**

No, a module cannot always refer to all variables declared in its parent module. Only variables declared as public in the parent module can be referred to by a module.

upvoted 1 times

 Sathisgm 1 year, 3 months ago

True we can refer parent module  
upvoted 1 times

 camps 1 year, 3 months ago

**Selected Answer: B**

False.

A module can refer to variables in its parent module only if they are explicitly passed to it using variable definitions or module blocks in the calling module.

upvoted 2 times

 Nunyabiznes 1 year, 3 months ago

**Selected Answer: B**

B. False

A module can only refer to variables that are explicitly passed to it as inputs, either through variable declarations or module calls. It cannot automatically refer to all variables declared in its parent module, as it may not need or have access to all of them. If a variable is not passed to the module as an input, the module will not be able to use it directly.

upvoted 1 times

 phidelics 1 year, 3 months ago

**Selected Answer: A**

refer to\* key word i guess  
upvoted 1 times

 rotimislaw 1 year, 4 months ago

**Selected Answer: A**

Seems A to me  
upvoted 1 times

 **agmesas** 1 year, 5 months ago

**Selected Answer: B**

B you must declare the variable in the child module if you can use it. Also, the parent module must call the child module using inputs. First, the parent module declares variables, second, the child module declares its inputs, then, the parent module calls to the child module using inputs that were declared in the variables.tf of the child module.

upvoted 3 times

 **Zeppoonstream** 1 year, 5 months ago

A. True. A module in Terraform can always refer to all variables declared in its parent module, as long as the parent module is called before the child module in the configuration file.

upvoted 1 times

 **ssanjayt** 1 year, 6 months ago

**Selected Answer: A**

Question #113

Topic 1

When you use a remote backend that needs authentication, HashiCorp recommends that you:

- A. Use partial configuration to load the authentication credentials outside of the Terraform code
- B. Push your Terraform configuration to an encrypted git repository
- C. Write the authentication credentials in the Terraform configuration files
- D. Keep the Terraform configuration files in a secret store

**Correct Answer: B**

*Community vote distribution*

A (100%)

 **Hizumi**  1 year, 10 months ago

Answer is A.

Note: We recommend omitting the token from the configuration, and instead using terraform login or manually configuring credentials in the C config file.

Reference: <https://www.terraform.io/language/settings/backends/remote>

upvoted 8 times

 **Bere**  7 months, 3 weeks ago

**Selected Answer: A**

Answer is A

This means that you should use environment variables to store these credentials. The partial configuration does not include sensitive data in the file but relies on environment variables that Terraform automatically detects.

Credentials and Sensitive Data

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#credentials-and-sensitive-data>

Warning: We recommend using environment variables to supply credentials and other sensitive data. If you use -backend-config or hardcode these values directly in your configuration, Terraform will include these values in both the .terraform subdirectory and in plan files. This can leak sensitive credentials.

upvoted 2 times

 **Tyler2023** 8 months, 2 weeks ago

Partial Configuration

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable if some arguments are provided automatically by an automation script running Terraform. When some or all of the arguments are omitted, we call this partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process.

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#partial-configuration>

upvoted 1 times

 **Tyler2023** 8 months, 2 weeks ago

Answer is A

upvoted 1 times

 **cruz95** 10 months, 1 week ago

**Selected Answer: A**

definitely

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: A**

it's a

upvoted 1 times

 **legendar7** 1 year, 9 months ago

A is the correct answer

upvoted 2 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: A**

<https://www.terraform.io/language/settings/backends/configuration#credentials-and-sensitive-data>

upvoted 2 times

 **jjkcoins** 1 year, 10 months ago

<https://www.terraform.io/language/settings/backends/configuration#credentials-and-sensitive-data>

upvoted 3 times

 **jjkcoins** 1 year, 10 months ago

Warning: We recommend using environment variables to supply credentials and other sensitive data. If you use -backend-config or hardcode these values directly in your configuration, Terraform will include these values in both the .terraform subdirectory and in plan files. This can leak sensitive credentials.

upvoted 3 times

 **hkacholiya** 1 year, 10 months ago

Answer is A

upvoted 1 times

 **shopkitty** 1 year, 10 months ago

**Selected Answer: A**

vote for A

upvoted 1 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: A**

It should be A.

upvoted 3 times

Question #114

Topic 1

You have a simple Terraform configuration containing one virtual machine (VM) in a cloud provider. You run `terraform apply` and the VM is created successfully.

What will happen if you `terraform apply` again immediately afterwards without changing any Terraform code?

- A. Terraform will terminate and recreate the VM
- B. Terraform will create another duplicate VM
- C. Terraform will apply the VM to the state file
- D. Nothing

**Correct Answer: C**

*Community vote distribution*

D (100%)

 **Bere** 7 months, 2 weeks ago

**Selected Answer: D**

If you run terraform apply again immediately afterwards without changing any Terraform code, Terraform will compare the desired state with the actual state of the infrastructure as recorded in the state file. Since there are no changes, then Terraform will report that there are no differences and won't do anything.

upvoted 4 times

 **Ni33** 1 year, 2 months ago

**Selected Answer: D**

D-Nothing

upvoted 1 times

 **Power123** 1 year, 3 months ago

Nothing. Answer is D

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: D**

keyword: Idempotent

upvoted 3 times

 **adouban** 1 year, 7 months ago

**Selected Answer: D**

Nothing

upvoted 1 times

 **legendary7** 1 year, 9 months ago

D is the correct answer. Terraform would refresh and see that the current state and the desired state are the-same and hence would do nothing.  
upvoted 2 times

 **shopkitty** 1 year, 10 months ago

**Selected Answer: D**

nothing change hence do nothing

upvoted 1 times

 **amanc** 1 year, 10 months ago

**Selected Answer: D**

Nothing.

If the desired state in state file is same as current state, no changes will happen

upvoted 4 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: D**

Nothing without changing the configuration file, the state file or desired state.

upvoted 3 times

A junior admin accidentally deleted some of your cloud instances. What does Terraform do when you run `terraform apply`?

- A. Build a completely brand new set of infrastructure
- B. Tear down the entire workspace infrastructure and rebuild it
- C. Rebuild only the instances that were deleted
- D. Stop and generate an error message about the missing instances

**Correct Answer: D**

*Community vote distribution*

C (90%)

10%

 **recep38** 4 months, 3 weeks ago

**Selected Answer: D**

it is very clear

upvoted 1 times

 **Ni33** 1 year, 2 months ago

**Selected Answer: C**

C is the correct answer !

upvoted 2 times

 **Power123** 1 year, 3 months ago

C is correct

upvoted 2 times

 **adouban** 1 year, 7 months ago

Deleting instance from cloud console will result terraform to create instance again to match what written in tf config files

upvoted 1 times

 **legendary7** 1 year, 9 months ago

C

Terraform would refresh and check that the Current State in the state file is not the same as the Desired State.

Terraform will check its configuration resource files and if it finds a resource that is present in the state file but not in the Desired state, it will create it

upvoted 4 times

 **shopkitty** 1 year, 10 months ago

**Selected Answer: C**

vote for C

upvoted 3 times

 **amamp** 1 year, 10 months ago

**Selected Answer: C**

Answer is C.

upvoted 2 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: C**

the correct answer is C.

upvoted 2 times

 **duffyduck** 1 year, 10 months ago

The Answer is C

upvoted 1 times

Question #116

*Topic 1*

You have created a main.tf Terraform configuration consisting of an application server, a database, and a load balancer. You ran `terraform apply` and all resources were created successfully. Now you realize that you do not actually need the load balancer so you run `terraform destroy` without any flags. What will happen?

- A. Terraform will destroy the application server because it is listed first in the code
- B. Terraform will prompt you to confirm that you want to destroy all the infrastructure
- C. Terraform will destroy the main.tf file
- D. Terraform will prompt you to pick which resource you want to destroy
- E. Terraform will immediately destroy all the infrastructure

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **SilentMilli** Highly Voted 1 year, 4 months ago

**Selected Answer: B**

Terraform will prompt you to confirm that you want to destroy all the resources before proceeding. The prompt will ask you to enter "yes" or "no" to confirm or cancel the destruction of resources unless you add the "-auto-approve" flag to stop terraform from prompting you to confirm.  
upvoted 6 times

 **ajbfffkabf** Most Recent 1 year ago

xxxxxxxxxxxxxx

upvoted 1 times

 **ozbeyucel** 1 year, 5 months ago

It is B, however, E is tricky there. can confuse  
upvoted 2 times

 **bora4motion** 1 year, 10 months ago

**Selected Answer: B**

I go with B

upvoted 3 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: B**

The answer is B.

upvoted 3 times

Question #117

Topic 1

Which type of block fetches or computes information for use elsewhere in a Terraform configuration?

- A. provider
- B. resource
- C. local
- D. data

**Correct Answer: A**

*Community vote distribution*

D (100%)

 **Burakko**  1 year, 10 months ago

**Selected Answer: D**

Data sources allow data to be fetched or computed for use elsewhere in Terraform configuration. Use of data sources allows a Terraform configuration to build on information defined outside of Terraform, or defined by another separate Terraform configuration.

upvoted 12 times

 **bora4motion**  1 year, 10 months ago

**Selected Answer: D**

it's D as you're using infrastructure already created - that's when you use "data". With "resource" you create new infrastructure.

upvoted 12 times

 **Ni33**  1 year, 2 months ago

**Selected Answer: D**

D is the correct answer !

upvoted 1 times

 **mkeology** 1 year, 3 months ago

**Selected Answer: D**

D. data, a data source

upvoted 1 times

 **Ravi528** 1 year, 3 months ago

**Selected Answer: D**

Definitely not A, It's D as Infra is done already, Data is used to compute

upvoted 2 times

 **SilentMilli** 1 year, 4 months ago

**Selected Answer: D**

The data block is used in Terraform to fetch or compute information from a given data source, and make that information available for use elsewhere in the Terraform configuration. This data can come from a variety of sources, including cloud provider APIs, databases, and other external systems.

upvoted 3 times

 **TechHero** 1 year, 5 months ago

D. data

Data blocks in Terraform are used to fetch or compute information for use elsewhere in a Terraform configuration. They allow you to read data from external sources such as an API, a file, or a database and use it to populate variables or other parts of your Terraform configuration.

A. provider

A provider block is used to configure the provider plugin used in Terraform configuration, it's not related to data fetching or computation

B. resource

A resource block is used to create and manage infrastructure resources, it's not related to data fetching or computation

C. local

A local block is used to define variables that are only used within the module in which they are defined and not exposed as outputs. It's not related to data fetching or computation.

It's important to understand that data blocks do not make changes to the infrastructure, they are only used to fetch or compute data.

upvoted 4 times

 **kprod** 1 year, 10 months ago

**Selected Answer: D**

D - data

upvoted 5 times

for the first time.

Which Terraform command should you run first?

- A. terraform apply
- B. terraform plan
- C. terraform show
- D. terraform init

**Correct Answer: C**

*Community vote distribution*

D (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: D**  
terraform init if you want to run it for the first time.  
upvoted 6 times

 **Pikopo** Most Recent 9 months, 2 weeks ago

D is correct  
upvoted 1 times

 **Ni33** 1 year, 2 months ago

**Selected Answer: D**  
If it is first time then Terraform init command.  
upvoted 1 times

 **Ravi528** 1 year, 3 months ago

**Selected Answer: D**  
First we should run terraform init  
upvoted 1 times

 **Manguu** 1 year, 3 months ago

**Selected Answer: D**  
This will initialize the working directory containing the TF config files  
upvoted 1 times

 **adouban** 1 year, 7 months ago

**Selected Answer: D**  
Terraform init  
upvoted 1 times

 **legendary7** 1 year, 9 months ago

D  
You need to initialize so all the provider plugin and configurations dependencies get initialized  
upvoted 1 times

 **kprod** 1 year, 10 months ago

**Selected Answer: D**  
D - init  
upvoted 3 times

 **bora4motion** 1 year, 10 months ago

**Selected Answer: D**  
I go with D. init  
upvoted 4 times

All modules published on the official Terraform Module Registry have been verified by HashiCorp.

- A. True
- B. False

**Correct Answer: B***Community vote distribution*

B (64%)

A (36%)

 **Hizumi** Highly Voted 1 year, 10 months ago

Answer is False.

Only modules considered "Verified Modules" are reviewed by Hashicorp, otherwise anyone can publish modules on the Terraform Registry.

Reference: <https://www.terraform.io/registry/modules/verified>

<https://www.terraform.io/registry/modules/publish>

upvoted 19 times

 **samimshaikh** Most Recent 6 months, 2 weeks ago

**Selected Answer: B**

<https://developer.hashicorp.com/terraform/registry/modules/verified>

upvoted 1 times

 **Tyler2023** 8 months, 2 weeks ago

False

Verified modules are expected to be actively maintained by HashiCorp partners. The verified badge isn't indicative of flexibility or feature support. Very simple modules can be verified just because they're great examples of modules. Likewise, an unverified module could be extremely high quality and actively maintained. An unverified module shouldn't be assumed to be poor quality, it only means it hasn't been created by a HashiCorp partner.

<https://developer.hashicorp.com/terraform/registry/modules/verified>

upvoted 1 times

 **Pikopo** 9 months, 2 weeks ago

B is correct

upvoted 1 times

 **NextBrand** 11 months, 3 weeks ago

B. False

A verified badge will usually appear on modules that are verified by Hashicorp. However, you can use filters to view unverified modules.  
upvoted 1 times

 **KJ\_Rollings** 1 year, 1 month ago

**Selected Answer: B**

<https://developer.hashicorp.com/terraform/registry/modules/verified>

upvoted 1 times

 **BennaniHaythem** 1 year, 3 months ago

True because When you submit your module, HashiCorp will review it to ensure that it meets the quality standards for the Terraform Module Registry. This review process includes checks for compliance with Terraform's best practices, security vulnerabilities, and other potential issues that could impact the stability or security of the infrastructure being provisioned with Terraform. If your module meets these standards and passes the verification process, it can be listed on the official Terraform Module Registry with the HashiCorp Verified badge.

upvoted 1 times

 **Ravi528** 1 year, 3 months ago

**Selected Answer: B**

False, It's not mandatory. If Hashicorp verified then it will show Blue tick at the end, Otherwise not. But it's not mandatory verification by hashicorp

upvoted 2 times

 **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: B**

B. False

While the Terraform Module Registry contains many useful modules, not all modules are verified by HashiCorp. Verified modules are indicated by a special badge on the module's page. These modules have been reviewed and approved by HashiCorp to ensure they follow best practices and are of high quality. Other modules in the registry are created and maintained by the community, and while they may be helpful, they have not gone through the same verification process. Always exercise caution and review the module's documentation and source code before using it in your infrastructure.

upvoted 1 times

 **rotimislaw** 1 year, 4 months ago

**Selected Answer: B**

Seems B to me

upvoted 1 times

 **Abuu** 1 year, 5 months ago

**Selected Answer: A**

The modules available on the official Terraform Module Registry have been reviewed by HashiCorp to ensure that they follow best practices and are compatible with Terraform. HashiCorp also provides support and guidance for users of the Module Registry.

upvoted 2 times

 **TechHero** 1 year, 5 months ago

B. False

The Terraform Module Registry is a public repository of modules created and shared by the community, it's not maintained by Hashicorp. These modules have not been verified by HashiCorp, and they may not have undergone any kind of security or quality review.

It's important to review the code and the documentation of the modules and also check if the author is trustworthy before using it in your infrastructure.

Although, Hashicorp provides a way to submit a module to the community and it goes through a review process to be added to the verified module list.

It's also important to check the module's version, release date, and the number of downloads before using it.

It's a good practice to use a module from a verified publisher or a module that is widely used and well-maintained.

It's also worth noting that you can use modules from other sources like GitHub, GitLab, or any other VCS. These modules may have been reviewed and validated by other organizations or communities, but they are not part of the official Terraform module registry.

upvoted 1 times

 **ssanjayt** 1 year, 6 months ago

**Selected Answer: A**

A is the correct one coz here its saying about the official hasi corp

upvoted 2 times

 **ArizonaClassics** 1 year, 7 months ago

A is the correct answer. Read: verified modules are reviewed by HashiCorp and actively maintained by contributors to stay up-to-date and compatible with Terraform and their respective providers.

The verified badge appears next to modules published by a verified source.

Verified modules are expected to be actively maintained by HashiCorp partners. The verified badge isn't indicative of flexibility or feature support; very simple modules can be verified just because they're great examples of modules. Likewise, an unverified module could be extremely high-quality and actively maintained. An unverified module shouldn't be assumed to be poor quality, it only means it hasn't been created by a HashiCorp partner.

<https://developer.hashicorp.com/terraform/registry/modules/verified>

upvoted 1 times

 **ArizonaClassics** 1 year, 7 months ago

I meant to SAY option B is the correct answer

upvoted 3 times

 **Bobby1977** 1 year, 9 months ago

<https://www.terraform.io/registry/modules/verified>

A is the right answer

upvoted 2 times

 **RVivek** 1 year, 9 months ago

Terraform public registry has community modules, which are not verified by Hashicorp

upvoted 2 times

 **Pinky0289** 1 year, 10 months ago

B. Not all modules published are verified by HashiCorp.

upvoted 2 times

Question #120

Topic 1

You have to initialize a Terraform backend before it can be configured.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (64%)

A (30%)

6%

 **yaza** Highly Voted 1 year, 9 months ago

Selected Answer: B

you cannot initialize something that does not exist, so obviously you have to configure it first

upvoted 25 times

 **zyxphreez** Highly Voted 1 year, 10 months ago

**Selected Answer: B**

you first configure the backed and then initialize ....therefore should be B

upvoted 10 times

 **RevSD** Most Recent 6 months, 1 week ago

In case of local backend , just an init is sufficient and no config is needed. May be question should be more clearer to say if it is local or remote

upvoted 2 times

 **samimshaikh** 6 months, 2 weeks ago

**Selected Answer: B**

B

When you change a backend's configuration, you must run terraform init again to validate and configure the backend before you can perform plans, applies, or state operations.

After you initialize, Terraform creates a .terraform/ directory locally. This directory contains the most recent backend configuration, including all authentication parameters you provided to the Terraform CLI. Do not check this directory into Git, as it may contain sensitive credentials for your remote backend.

upvoted 2 times

 **shortcut** 8 months ago

**Selected Answer: B**

I guess, here "initialize" means "instantiate" i.e. the backend must be there For e.g.- a S3 bucket should be present first before we can use it (configure it in HCL). So, from that perspective, it should be "A". However, we can also have local backends, which is where the terraform configuration file themselves reside, and we don't have to separately instantiate those. Hence, the answer is "B"

upvoted 1 times

 **aanataliya** 10 months ago

**Selected Answer: A**

Configure doesn't mean writing configuration of "backend block" in this context. Hashicorp always use "block" if they want to mention writing configuration. example : backend block, terraform block or cloud block. If I say, I want to configure linux server doesn't mean writing HCL for I server. Just writing "backend block" doesn't actually configure backend. You have to run/execute that HCL "backend block" to actually create backend configuration.

During actual backend configuration, Terraform writes the backend configuration in plain text in two separate files.

Check Ref: <https://developer.hashicorp.com/terraform/language/settings/backends/configuration>

And this cannot be done until we first execute init.

upvoted 1 times

 **3cc17f1** 8 months, 3 weeks ago

That link you provided contradicts your answer:

"When you change a backend's configuration, you must run terraform init again to validate and configure the backend before you can perform any plans, applies, or state operations."

Clearly this suggests that configuration occurs PRIOR to initialisation.

upvoted 3 times

 **arunkaushik** 11 months, 1 week ago

Correct is B:

How sleepy are people who voted for an option A... like compiling the code before writing it.

upvoted 2 times

 **milan92stankovic** 1 year, 1 month ago

**Selected Answer: B**

I vote for B

upvoted 2 times

 **Priyankakanna** 1 year, 1 month ago

What I understand is who choose false confusing with terminology its config nor writing configuration ie before creating the resource ie apply need to initialise the backend so we have to run init so that from the configuration files it will download the necessary plugins for the backend then apply the configuration to configure the resources

upvoted 3 times

 **piezas** 1 year, 1 month ago

**Selected Answer: U**

A. True

Before configuring and using a Terraform backend, you need to initialize it first. The initialization process sets up the backend configuration, such as the type of backend (e.g., S3, Azure Storage, etc.) and the connection details. This is done by running the terraform init command, which initializes the working directory and prepares it for Terraform operations, including the configuration of the backend.

upvoted 3 times

 **sdm13168** 1 year, 1 month ago

**Selected Answer: B**

B you need to configure the backend first, then run terraform init

upvoted 1 times

 **kiran15789** 1 year, 2 months ago

**Selected Answer: A**

Before configuring a backend in Terraform, it is necessary to initialize it. This is done by running the terraform init command, which sets up the working directory for the configuration, downloads any necessary plugins, and initializes the backend. Once the backend is initialized, you can configure it by editing the backend block in the Terraform configuration file.

upvoted 1 times

 **Ni33** 1 year, 2 months ago

**Selected Answer: B**

B is the correct answer !

upvoted 1 times

 Stanislav4907 1 year, 3 months ago

**Selected Answer: A**

Yes, before configuring a Terraform backend, you need to initialize it. Initializing a backend sets up the necessary resources and configuration for a particular backend, such as creating storage containers or initializing remote state.

To initialize a backend, you can use the `terraform init` command. This command reads the configuration files in your working directory and installs any necessary plugins or modules, as well as initializes the backend you have specified in your configuration.

After initializing the backend, you can configure it by adding the appropriate configuration to your Terraform configuration files. The backend configuration typically includes details such as the backend type, the backend's credentials or access details, and any additional settings specific to the backend you're using.

Once you have initialized and configured your backend, you can then use Terraform to create and manage infrastructure resources in the backend.

upvoted 2 times

 Nunyabiznes 1 year, 3 months ago

**Selected Answer: B**

B. False

The backend configuration is specified within the Terraform configuration files, typically in the `main.tf` file. You don't have to initialize a backend before configuring it; instead, you provide the backend configuration within the Terraform files, and then run `terraform init`. The initialization process will set up the backend as per the configuration provided. So, the backend configuration is done first, and then you initialize Terraform which sets up the backend accordingly.

upvoted 4 times

 \_sreya23\_ 1 year, 3 months ago

**Selected Answer: B**

Without configuration (not writing code) you can't initialize it.

upvoted 1 times

 khaled\_razouk 1 year, 4 months ago

**Selected Answer: B**

you can't do that if it doesn't exist

upvoted 1 times

Question #121

Topic 1

Which of the following does terraform apply change after you approve the execution plan? (Choose two.)

- A. Cloud infrastructure
- B. The `.terraform` directory
- C. The execution plan
- D. State file
- E. Terraform code

**Correct Answer: C**

*Community vote distribution*

A (66%)

D (34%)

 **faltu1985** Highly Voted 1 year, 9 months ago

**Selected Answer: A**

A and D

upvoted 24 times

 **camps** Highly Voted 1 year, 3 months ago

After you approve the execution plan, terraform apply will make changes to both the cloud infrastructure and the state file. Therefore, options and D are correct. The .terraform directory is a local directory used by Terraform to store plugins and other files, and is not modified by terraform apply. The execution plan is generated by Terraform to show you the changes that will be made, and is not modified by terraform apply. The Terraform code is also not modified by terraform apply; it simply tells Terraform what changes to make to the infrastructure.

upvoted 10 times

 **f2e2419** Most Recent 5 months, 3 weeks ago

**Selected Answer: D**

A and D

upvoted 3 times

 **[Removed]** 7 months ago

**Selected Answer: D**

A and D

upvoted 3 times

 **awsexams** 7 months ago

A and D

upvoted 3 times

 **Selva247** 8 months, 1 week ago

A. Cloud infrastructure: Terraform applies the changes to your cloud infrastructure based on the configuration described in your Terraform config file.

D. State file: Terraform applies the changes and updates the state file to reflect the current state of the infrastructure.

upvoted 3 times

 **Pikopo** 9 months, 2 weeks ago

A and D

upvoted 2 times

 **VSMu** 12 months ago

**Selected Answer: A**

A and D.

upvoted 1 times

 **abhi6199** 1 year ago

A and D are the correct options.

upvoted 1 times

 **March2023** 1 year, 1 month ago

**Selected Answer: A**

A and D for sure.

upvoted 1 times

 **KJ\_Rollings** 1 year, 1 month ago

**Selected Answer: D**

A and D

upvoted 2 times

 **Ni33** 1 year, 2 months ago

A is the correct answer.

upvoted 1 times

 **wojtec** 1 year, 2 months ago

A and D

upvoted 2 times

 **Ravi528** 1 year, 3 months ago

**Selected Answer: A**

Cloud infra will be updated/changed not the execution plan.

It will update the content/values in state file based on latest changes happened as a part of apply.

I will go with A

upvoted 1 times

 **SilentMilli** 1 year, 4 months ago

**Selected Answer: A**

When you run terraform apply and approve the execution plan, Terraform will make changes to the cloud infrastructure based on the desired state defined in the Terraform code. Terraform will also update the state file to reflect the new state of the infrastructure after the changes have been made. Therefore, options A and D are correct.

upvoted 1 times

 **rotimislaw** 1 year, 4 months ago

**Selected Answer: D**

A and D

upvoted 3 times

 **mpetbteam** 1 year, 4 months ago

I think the question is bad formulated, not all providers are cloud-based

upvoted 1 times

Question #122

Topic 1

A Terraform backend determines how Terraform loads state and stores updates when you execute \_\_\_\_\_.

- A. apply
- B. taint
- C. destroy
- D. All of the above
- E. None of the above

**Correct Answer: E**

*Community vote distribution*

D (100%)

 **amamp** Highly Voted 1 year, 10 months ago

**Selected Answer: D**

D is right

upvoted 7 times

 **imkhan** Most Recent 9 months ago

D. All of the above

Explanation: A Terraform backend is responsible for managing the state of your infrastructure and how Terraform loads and stores updates for various operations, including apply, taint, and destroy, as well as other commands like plan, import, and more. The backend defines how and where the Terraform state is stored and retrieved, and it's used in various Terraform operations to ensure consistency and manage the state of your infrastructure.

upvoted 3 times

 **Pikopo** 9 months, 2 weeks ago

D is right

upvoted 1 times

 **Foram31** 1 year ago

**Selected Answer: D**

When you execute commands like terraform apply, terraform taint, or terraform destroy, Terraform needs to access and update the state file to perform the desired actions. The backend determines how Terraform interacts with the state file during these operations.

upvoted 3 times

 **Ni33** 1 year, 2 months ago

**Selected Answer: D**

D is correct

upvoted 1 times

 **BennaniHaythem** 1 year, 3 months ago

Select Answer :A apply.

The taint and destroy commands do not involve loading or storing state, but can interact with state data depending on how they are used.

upvoted 2 times

 **Halimb** 10 months, 1 week ago

Upvoted by mistake; this is wrong. Correct answer is C.

upvoted 1 times

 **Pinky0289** 1 year, 10 months ago

D is the answer

upvoted 1 times

 **harshit101** 1 year, 10 months ago

Selected Answer: D

D is correct

upvoted 1 times

 **bora4motion** 1 year, 10 months ago

**Selected Answer: D**

D looks OK

upvoted 3 times

 **dc\_98** 1 year, 10 months ago

**Selected Answer: D**

Thats when state is used so the backend should be looked at

upvoted 4 times

- A. Tracking provider dependencies
- B. There is no such file
- C. Preventing Terraform runs from occurring
- D. Storing references to workspaces which are locked

**Correct Answer:** D

*Community vote distribution*

A (96%)

4%

 [Removed]  1 year, 10 months ago

**Selected Answer: A**

A is correct

<https://www.terraform.io/language/files/dependency-lock>

upvoted 9 times

 Nunyabiznes  1 year, 3 months ago

**Selected Answer: A**

A. Tracking provider dependencies

The .terraform.lock.hcl file is used to track provider dependencies and their exact versions. This file is automatically generated by Terraform when you run terraform init. It locks the versions of the providers used in your configuration, ensuring that subsequent Terraform runs use the same provider versions for consistency and reproducibility across environments. This file should be committed to your version control system to maintain consistency across team members and environments.

upvoted 7 times

 Pikopo  9 months, 2 weeks ago

A is correct

upvoted 1 times

 Ni33 1 year, 2 months ago

**Selected Answer: A**

~~A is the correct answer!~~

Question #124

*Topic 1*

You've used Terraform to deploy a virtual machine and a database. You want to replace this virtual machine instance with an identical one without affecting the database. What is the best way to achieve this using Terraform?

- A. Use the terraform state rm command to remove the VM from state file
- B. Use the terraform taint command targeting the VMs then run terraform plan and terraform apply
- C. Use the terraform apply command targeting the VM resources only
- D. Delete the Terraform VM resources from your Terraform code then run terraform plan and terraform apply

**Correct Answer: B**

*Community vote distribution*

B (100%)

✉  **KJ\_Rollings** Highly Voted 1 year, 1 month ago

**Selected Answer: B**

B. for v0.15.2 and later, you can also use "terraform apply -replace=ADDRESS" to achieve the same result.  
upvoted 7 times

✉  **keiffo2** Highly Voted 1 year, 10 months ago

terraform apply -target - just looks at applying part of the config.

In this case we'd need to taint the resource and then run an apply. Just to note taint is now deprecated - seems like it comes up in the exam mind, instead of using taint you should use terraform apply -replace=aws\_instance.my\_server

upvoted 5 times

✉  **alirasouli** 1 year, 7 months ago

Exactly I was thinking about `terraform apply -replace=...`, so answer C can also be correct. Of course answer B is also correct and deprecated!

upvoted 1 times

✉  **keiffo2** 1 year, 10 months ago

So answer B

upvoted 2 times

✉  **Spandrop** 8 months ago

being taint deprecated, I would go with C.

upvoted 3 times

✉  **junk4share** Most Recent 1 year ago

**Selected Answer: B**

ITS B.

upvoted 1 times

Question #125

Topic 1

How do you specify a module's version when publishing it to the public Terraform Module Registry?

- A. The module's configuration page on the Terraform Module Registry
- B. Terraform Module Registry does not support versioning modules
- C. The release tags in the associated repo
- D. The module's Terraform code

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

The registry uses tags to identify module versions. Release tag names must be a semantic version, which can optionally be prefixed with a v. For example, v1. 0.4 and 0.9.

upvoted 9 times

 **Tyler2023** Most Recent 8 months, 2 weeks ago

answer is C:

Once a module is published, you can release a new version of a module by simply pushing a properly formed Git tag.

x.y.z tags for releases. The registry uses tags to identify module versions. Release tag names must be a semantic version, which can optionally be prefixed with a v. For example, v1.0.4 and 0.9.2. To publish a module initially, at least one release tag must be present. Tags that don't look like version numbers are ignored.

<https://developer.hashicorp.com/terraform/registry/modules/publish>

upvoted 3 times

 **Ni33** 1 year, 2 months ago

**Selected Answer: C**

C is the correct answer.

upvoted 1 times

 **SilentMilli** 1 year, 4 months ago

**Selected Answer: C**

C. The release tags in the associated repo. When publishing a module to the Terraform Module Registry, the module version is typically specified using Git tags in the associated Git repository. Terraform retrieves the module from the registry using the provider's source argument, which includes the module's owner, name, and version. By specifying the version in the source argument of your module, Terraform will download the specific version of the module from the registry.

upvoted 1 times

 **Pinky0289** 1 year, 10 months ago

A module's version is specified using the release tag names. So, the right answer is C.

upvoted 2 times

 **keiffo2** 1 year, 10 months ago

x.y.z tags for releases. The registry uses tags to identify module versions. Release tag names must be a semantic version, which can optionally be prefixed with a v. For example, v1.0.4 and 0.9.2. To publish a module initially, at least one release tag must be present. Tags that don't look like version numbers are ignored.

ANSWER C

upvoted 3 times

Question #126

Topic 1

Terraform plan updates your state file.

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

B (67%)

A (33%)

 **Nunyabiznes** Highly Voted 1 year, 3 months ago

**Selected Answer: B**

B. False

The terraform plan command does not update your state file. Instead, it generates an execution plan by comparing the desired state (defined by your Terraform configuration files) with the actual state (stored in the state file) of your infrastructure. The plan shows the changes that Terraform will make to the infrastructure without actually applying those changes. The state file is only updated when you run terraform apply, which applies the changes to your infrastructure and updates the state file to reflect the new state.

upvoted 14 times

✉️  **Halimb** 10 months, 1 week ago

A good explanation. If anyone is still doubting this; remember that the state file reflects your actual infrastructure. Since "terraform plan" does not actually change your infrastructure, it would make no sense for it to change your state file. That would make your state file invalid (state drift)!

upvoted 4 times

✉️  **RVivek**  1 year, 9 months ago

**Selected Answer: A**

Terraform plan always first check the state of existing resources refreshes/updates that to state.

If there is configuration drift from last apply, then terraform will first refresh the state to resreset the current state of the resources, for example any resource is deleted or changed that will be upadted in the state and then the chnages to be doe will be displayed

upvoted 7 times

✉️  **RonZhong** 11 months, 1 week ago

Yes, everytime when I run Terraform Plan, my backend state file is also updated. I can see the last modified timestamp has been updated, even I didn't apply the change.

upvoted 2 times

✉️  **AkaAka4**  2 months, 2 weeks ago

**Selected Answer: A**

Refer to <https://stackoverflow.com/questions/66240556/terraform-plan-not-updating-state-file>

upvoted 1 times

✉️  **AkaAka4** 2 months, 2 weeks ago

Check the 2nd last comment in the link, should be B. Apologies.

upvoted 1 times

✉️  **Ajit18** 4 months ago

B. False

Terraform plan does not update the state file itself. It performs an in-memory refresh to reconcile the state file with the actual infrastructure, but doesn't modify the state file on disk. The update to the state file happens during terraform apply if there are changes required.

upvoted 1 times

✉️  **dizzy\_monkey** 4 months, 1 week ago

**Selected Answer: A**

"A" should be correct :

"Terraform automatically performs the same refreshing actions as a part of creating a plan in both the terraform plan and terraform apply commands."

<https://developer.hashicorp.com/terraform/cli/commands/refresh>

upvoted 1 times

✉️  **91576b8** 5 months ago

**Selected Answer: A**

when you run terraform plan , it will refresh the state file automatically

upvoted 1 times

✉️  **boapaulo** 5 months, 2 weeks ago

The terraform plan command does not update the state file. It performs a refresh of the state in-memory to generate an execution plan, but it does not persist these changes to the state file. The state file is updated when you run terraform apply.

upvoted 1 times

✉️  **vipulchoubisa** 6 months, 1 week ago

create one simple instance and change the tags from AWS console. After this run terraform plan command. As a result it wont update the state file with new tags. So answer is B

upvoted 1 times

✉️  **Demonik** 7 months, 1 week ago

**Selected Answer: B**

Nope.Plan does not change state file.But others like apply destroy after approve will :)

upvoted 1 times

 **pamfidelis** 9 months, 2 weeks ago

**Selected Answer: A**

A. True

"You can also update your state file without making modifications to your infrastructure using the -refresh-only flag for plan and apply operations."

<https://developer.hashicorp.com/terraform/tutorials/state/refresh>

upvoted 1 times

 **TigerInTheCloud** 6 months, 3 weeks ago

terraform plan -refresh-only only make the plan of changing state file.

upvoted 1 times

 **gofavad926** 9 months, 2 weeks ago

**Selected Answer: B**

B, terraform plan doesn't update the state

upvoted 1 times

 **nahed** 10 months, 2 weeks ago

**Selected Answer: A**

terraform state will UPDATE your state file in case you change the resources outside the terraform. IT will "sync" your state file with the actual infrastructure.

upvoted 1 times

 **kudakk** 11 months ago

**Selected Answer: B**

The terraform plan command does not change the state of your infrastructure. It creates an execution plan that lets you preview the changes Terraform plans to make to your infrastructure

upvoted 1 times

 **foreverlearner** 1 year ago

"To determine whether state drift occurred, Terraform performs a refresh operation before it begins to build an execution plan. This refresh step pulls the actual state of all of the resources currently tracked in your state file. --> Terraform does not update your actual state file <--, but captures the refreshed state in the plan file." (<https://developer.hashicorp.com/terraform/tutorials/cli/plan>)

"You can also update your state file without making modifications to your infrastructure using the -refresh-only flag for plan and apply operations." (<https://developer.hashicorp.com/terraform/tutorials/state/refresh>)

So plan doesn't update the state file (just an in-memory sync) unless you specify the --refresh-only flag

upvoted 5 times

 **milan92stankovic** 1 year, 1 month ago

**Selected Answer: A**

terraform state will UPDATE your state file in case you change the resources outside the terraform. IT will "sync" your state file with the actual infrastructure.

upvoted 4 times

 **Ni33** 1 year, 2 months ago

**Selected Answer: B**

B is the correct answer

upvoted 1 times

 **ale\_brd\_** 1 year, 5 months ago

**Selected Answer: B**

b is the correct answer

upvoted 1 times

- A. terraform fmt -check
- B. terraform fmt -write=false
- C. terraform fmt --list -recursive
- D. terraform fmt -check -recursive

**Correct Answer: C**

*Community vote distribution*

D (100%)

✉  **RVivek** Highly Voted 1 year, 9 months ago

**Selected Answer: D**

terraform fmt takes followong options

Options:

-list=false Don't list files whose formatting differs  
(always disabled if using STDIN)

-write=false Don't write to source files  
(always disabled if using STDIN or -check)

-diff Display diffs of formatting changes

-check Check if the input is formatted. Exit status will be 0 if all  
input is properly formatted and non-zero otherwise.

-no-color If specified, output won't contain any color.

-recursive Also process files in subdirectories. By default, only the  
given directory (or current directory) is processed.

upvoted 10 times

✉  **jabbawockie12** Highly Voted 1 year, 10 months ago

**Selected Answer: D**

-check -recursive should check all formatting without modifying the configuration files.

upvoted 8 times

✉  **Anderson01** Most Recent 1 year, 8 months ago

"fmt -> without making changes"

So, I will go with D (make sure)

upvoted 1 times

✉  **Pinky0289** 1 year, 10 months ago

terraform fmt -check -recursive, recursively checks all modules and submodules for the format.

upvoted 2 times

✉  **Pinky0289** 1 year, 10 months ago

The answer is C.

upvoted 1 times

✉  **ptR95** 1 year, 9 months ago

The answer is D.

upvoted 2 times

✉  **zyxphreez** 1 year, 10 months ago

**Selected Answer: D**

-check Check if the input is formatted. Exit status will be 0 if all  
input is properly formatted and non-zero otherwise.

-recursive Also process files in subdirectories. By default, only the  
given directory (or current directory) is processed.

answer for me is D

upvoted 4 times

Question #128

Topic 1

As a member of the operations team, you need to run a script on a virtual machine created by Terraform. Which provision is best to use in your Terraform code?

- A. null-exe $\mu$
- B. local-exec
- C. remote-exec
- D. file

**Correct Answer: B**

*Community vote distribution*

C (100%)

RVivek Highly Voted 1 year, 9 months ago

Selected Answer: C

remote-exec to run anything on the deployed virtual machine.  
local exec is to run anything on the system where terraform is running  
upvoted 21 times

bluee 1 year, 6 months ago

thanks mate  
upvoted 3 times

Uma10 Highly Voted 1 year, 10 months ago

Selected Answer: C

The remote-exec provisioner invokes a script on a remote resource after it is created.

To invoke a local process, see the local-exec provisioner instead.

Source: <https://www.terraform.io/language/resources/provisioners/remote-exec>

upvoted 9 times

Question #129

Topic 1

You are using a networking module in your Terraform configuration with the name label my\_network. In your main configuration you have the following code:

```
output: "net_id" {
 value = module.my_network.vnet_id
}
```

When you run terraform validate, you get the following error:

```
Error: Reference to undeclared output value

on main.tf line 12, in output "net_id":
12: value = module.my_network.vnet_id
```

What must you do to successfully retrieve this value from your networking module?

- A. Define the attribute vnet\_id as a variable in the networking module
- B. Change the referenced value to module.my\_network.outputs.vnet\_id
- C. Define the attribute vnet\_id as an output in the networking module
- D. Change the referenced value to my\_network.outputs.vnet\_id

Correct Answer: D

Community vote distribution

C (93%)

7%

 **zyxphreez** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

In a parent module, outputs of child modules are available in expressions as module.<MODULE NAME>.<OUTPUT NAME>. For example, if a child module named web\_server declared an output named instance\_ip\_addr, you could access that value as module.web\_server.instance\_ip\_addr.

Answer is C

upvoted 19 times

 **Bere** Most Recent 6 months ago

**Selected Answer: A**

Answer is A

In Terraform versions 0.11 and earlier, you would use the interpolation syntax with \${} to reference attributes:

```
Terraform 0.11 and earlier
output "instance_ip_addr" {
 value = "${aws_instance.main.private_ip}"
}
```

Starting with Terraform 0.12, interpolation is not needed for simple references, so you can directly reference the attribute:

```
Terraform 0.12 and later
output "instance_ip_addr" {
 value = aws_instance.main.private_ip
}
```

upvoted 2 times

 **Bere** 6 months ago

**Selected Answer: C**

Answer is C

Define the attribute vnet\_id as an output in the networking module.

In the Child Module (my\_network), if you had a resource defined as aws\_vpc and named it example\_vpc, you would reference its ID like this:

# In your networking module's outputs.tf or main.tf file

```
output "vnet_id" {
 description = "The ID of the created VNet"
 value = aws_vpc.example_vpc.id # Replace example_vpc with the actual name of your VPC resource
}
```

In the Parent Module (main configuration):

```
output "net_id" {
 value = module.my_network.vnet_id
}
```

upvoted 2 times

 **Jayanth** 11 months, 3 weeks ago

C is the right answer

upvoted 1 times

 **Williamus** 1 year, 4 months ago

**Selected Answer: C**

Question #130

Topic 1

You are writing a child Terraform module which provisions an AWS instance. You want to make use of the IP address returned in the root configuration. You name the instance resource "main".

Which of these is the correct way to define the output value using HCL2?

A.

```
output "instance_ip_addr" {
 value = "${aws_instance.main.private_ip}"
}
```

B.

```
output "instance_ip_addr" {
 return aws_instance.main.private_ip
}
```

**Correct Answer: A**

✉ keiffo2 Highly Voted 1 year, 10 months ago

Has to be Answer A. no such definition as "return"

upvoted 21 times

✉ Bere Most Recent 6 months ago

Answer is A

In Terraform versions 0.11 and earlier, you would use the interpolation syntax with \${} to reference attributes:

```
Terraform 0.11 and earlier
output "instance_ip_addr" {
 value = "${aws_instance.main.private_ip}"
}
```

Starting with Terraform 0.12, interpolation is not needed for simple references, so you can directly reference the attribute:

```
Terraform 0.12 and later
output "instance_ip_addr" {
 value = aws_instance.main.private_ip
}
```

upvoted 2 times

✉ MauroSoli 8 months, 3 weeks ago

Answer is A but syntax is deprecated

upvoted 3 times

✉ syam22587 10 months, 3 weeks ago

A is correct answer

upvoted 1 times

✉ junk4share 1 year ago

A is correct.

upvoted 1 times

✉ phidelics 1 year, 3 months ago

The answer is A

upvoted 1 times

✉ kennynelcon 1 year, 5 months ago

A is accurate

upvoted 1 times

Question #131

Topic 1

How can a ticket-based system slow down infrastructure provisioning and limit the ability to scale? (Choose two.)

- A. A full audit trail of the request and fulfillment process is generated
- B. A request must be submitted for infrastructure changes
- C. As additional resources are required, more tickets are submitted
- D. A catalog of approved resources can be accessed from drop down lists in a request form

**Correct Answer: B**

*Community vote distribution*

C (61%)

B (39%)

 **Darshan07** 4 months, 2 weeks ago

**Selected Answer: C**

Both B & C

upvoted 1 times

 **NashP** 5 months, 1 week ago

- B. A request must be submitted for infrastructure changes
- C. As additional resources are required, more tickets are submitted

Explanation:

B. A request must be submitted for infrastructure changes: In a ticket-based system, users often need to raise a request or a ticket for any infrastructure changes. This process can introduce delays, especially when changes are needed urgently or frequently. It can slow down the provisioning process as it adds an additional layer of approval and manual intervention.

C. As additional resources are required, more tickets are submitted: In a ticket-based system, when scaling is required, users may need to submit additional tickets for each resource or instance they need. This manual process can become a bottleneck as the number of tickets increases, making it less efficient for scaling rapidly or handling dynamic workloads.

upvoted 2 times

 **[Removed]** 6 months, 3 weeks ago

**Selected Answer: C**

B and C

upvoted 1 times

 **[Removed]** 7 months ago

**Selected Answer: C**

B and C. increase amount of C

upvoted 1 times

 **kennynelcon** 1 year, 5 months ago

**Selected Answer: C**

B and C

lets populate C

upvoted 3 times

 **Edileimig** 1 year, 6 months ago

**Selected Answer: C**

B and C

upvoted 1 times

 **robertninho** 1 year, 6 months ago

B and C

upvoted 1 times

 **DawidB** 1 year, 6 months ago

**Selected Answer: C**

B and C indeed.

upvoted 1 times

 **legendary7** 1 year, 9 months ago

B and C are correct

upvoted 3 times

 **dani88ge** 1 year, 9 months ago

**Selected Answer: C**

B and C

upvoted 3 times

 **Pinky0289** 1 year, 10 months ago

B and C

upvoted 2 times

 **dinesh198728** 1 year, 10 months ago

**Selected Answer: B**

B and C both ticket based

upvoted 3 times

 **kprod** 1 year, 10 months ago

**Selected Answer: B**

Answer is BC

upvoted 4 times

Question #132

Topic 1

Which of the following statements about Terraform modules is not true?

- A. Modules must be publicly accessible
- B. Modules can be called multiple times
- C. Module is a container for one or more resources
- D. Modules can call other modules

**Correct Answer: C**

*Community vote distribution*

A (100%)

 **Uma10** Highly Voted 1 year, 10 months ago

**Selected Answer: A**

A: In addition to modules from the local filesystem, Terraform can load modules from a public or private registry. Also, members of your organization might produce modules specifically crafted for your own infrastructure needs.

Source: <https://www.terraform.io/language/modules>

upvoted 16 times

 **ravi135** Most Recent 5 months, 1 week ago

A is right

upvoted 1 times

 **Daro\_** 1 year, 5 months ago

**Selected Answer: A**

A is correct

upvoted 1 times

 **ruganesh** 1 year, 5 months ago

A is the correct one.

upvoted 1 times

 **adouban** 1 year, 7 months ago

**Selected Answer: A**

A is the correct

upvoted 1 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: A**

Modules can be from local disk , private registry in your company private network

upvoted 2 times

 **Pinky0289** 1 year, 10 months ago

Besides being publicly accessible, modules can also be privately accessible only within the organization. So, the answer is option A.

upvoted 4 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: A**

It can be private modules too...

upvoted 4 times

Question #133

Topic 1

Which Terraform collection type should you use to store key/value pairs?

A. tuple

B. set

C. map

D. list

**Correct Answer: C**

Community vote distribution

C (100%)

✉️  **Uma10** Highly Voted 1 year, 10 months ago

The answer is correct.

Maps/objects are represented by a pair of curly braces containing a series of <KEY> = <VALUE> pairs

Source: <https://www.terraform.io/language/expressions/types>

upvoted 7 times

✉️  **debabrata6983** Most Recent 10 months, 3 weeks ago

Selected Answer: C

map collection

upvoted 1 times

✉️  **camps** 1 year, 3 months ago

Selected Answer: C

C. map is the Terraform collection type that should be used to store key/value pairs.

A map is an unordered collection of key/value pairs, where each key is unique. It can be used to store related data that doesn't have a specific order, and it's useful for storing configuration parameters or metadata associated with infrastructure resources. A tuple is an ordered collection of elements, a set is an unordered collection of unique elements, and a list is an ordered collection of elements that can include duplicates. While each of these collection types has its own uses, a map is the most appropriate type for storing key/value pairs.

upvoted 3 times

✉️  **Pinky0289** 1 year, 10 months ago

Elements stored in key=value pairs within curly braces is a map. So, the right option is C.

upvoted 1 times

✉️  **dепал\_dhir** 1 year, 10 months ago

Selected Answer: C

C - Map is the correct answer

<https://www.terraform.io/language/expressions/types>

upvoted 1 times

✉️  **keiffo2** 1 year, 10 months ago

map variable is key-value

upvoted 1 times

✉️  **keiffo2** 1 year, 10 months ago

so C is the right answer

upvoted 1 times

You have used Terraform to create an ephemeral development environment in the cloud and are now ready to destroy all the infrastructure described by your Terraform configuration. To be safe, you would like to first see all the infrastructure that will be deleted by Terraform. Which command should you use to show all of the resources that will be deleted? (Choose two.)

- A. Run terraform plan -destroy
- B. Run terraform show -destroy
- C. Run terraform destroy and it will first output all the resources that will be deleted before prompting for approval
- D. Run terraform show -destroy

**Correct Answer:** AD

*Community vote distribution*

AC (100%)

 **bora4motion** Highly Voted 1 year, 10 months ago

**Selected Answer:** AC

I go with AC

upvoted 9 times

 **imkhan** Most Recent 9 months ago

AC are the correct options

When you want to see a preview of the resources that will be deleted by Terraform, you can use the terraform plan -destroy command to generate a destruction plan that shows what will be removed. This is a safe way to ensure you understand the impact of the destroy operator before proceeding.

Option C is also correct because when you run terraform destroy, Terraform will provide an execution plan that lists the resources to be deleted before it actually prompts you for approval to proceed with the destruction. This allows you to review the list of resources that will be affected before confirming the destroy operation.

upvoted 4 times

 **gofavad926** 9 months, 2 weeks ago

**Selected Answer:** AC

AC for sure

upvoted 1 times

 **BilalIg93350** 1 year, 4 months ago

A. Run terraform plan -destroy

D. Run terraform show -destroy

To see all the resources that will be deleted by Terraform, you should run the terraform plan -destroy command. This command will show a preview of all the changes that Terraform will make, including any resources that will be deleted.

After running the terraform plan -destroy command, you can also run the terraform show -destroy command to view a detailed summary of the resources that will be deleted.

Option C (Run terraform destroy and it will first output all the resources that will be deleted before prompting for approval) is not correct. Run the terraform destroy command will immediately destroy all the resources without showing a preview of what will be deleted. To be safe, it is recommended to run the terraform plan -destroy command first to see the resources that will be deleted, and then run terraform destroy to confirm the deletion.

upvoted 4 times

 **MrTee** 12 months ago

Terraform show -destroy is not a valid command

upvoted 4 times

 **Roytf** 9 months, 1 week ago

don't paste chatGpt output..option D is invalid

upvoted 3 times

 **Halimb** 10 months, 1 week ago

Nonsense. There is no terraform show -destroy. Stop confusing people. Answer is A&C.

upvoted 4 times

 **VamsiPopuri** 1 year, 5 months ago

**Selected Answer: AC**

A and C.

Aren't B and D the same ?

upvoted 4 times

 **rafpe** 1 year, 6 months ago

**Selected Answer: AC**

Def A&C

upvoted 1 times

 **Atta33** 1 year, 9 months ago

**Selected Answer: AC**

AC is correct

upvoted 3 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: AC**

A and C

upvoted 1 times

 **shopkitty** 1 year, 10 months ago

**Selected Answer: AC**

vote for A & C

upvoted 1 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: AC**

It is AC. terraform show only shows the state file

upvoted 4 times

 **dc\_98** 1 year, 10 months ago

**Selected Answer: AC**

There is no terraform show -destroy

upvoted 3 times

When do you need to explicitly execute terraform refresh?

- A. Before every terraform plan
- B. Before every terraform apply
- C. Before every terraform import
- D. None of the above

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Uma10** Highly Voted 1 year, 10 months ago

The answer is correct.

Wherever possible, avoid using terraform refresh explicitly and instead rely on Terraform's behavior of automatically refreshing existing objects part of creating a normal plan.

Source: <https://www.terraform.io/cli/commands/refresh>  
upvoted 10 times

 **keiffo2** 1 year, 10 months ago

so the answer is D  
upvoted 2 times

 **Darshan07** Most Recent 4 months, 2 weeks ago

**Selected Answer: D**

D is correct  
upvoted 1 times

 **hajurbau** 10 months, 1 week ago

**Selected Answer: D**

D is correct  
upvoted 1 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: D**

As per Terraform documentation refresh command is not recommended, it is there only for backward compatibility  
upvoted 2 times

 **Pinky0289** 1 year, 10 months ago

D. None of the above.

Refresh is used to check backward compatibility to read the infrastructure objects, and update the state file. Terraform performs refresh by its during every plan and apply commands. So, it is recommended that we not perform refresh often.  
upvoted 2 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: D**

<https://www.terraform.io/cli/commands/refresh>  
upvoted 1 times

All Terraform Cloud tiers support team management and governance.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

B (91%)

9%

 **Uma10** Highly Voted 1 year, 10 months ago

**Selected Answer: B**

Terraform Cloud is a commercial SaaS product developed by HashiCorp. Many of its features are free for small teams, including remote state storage, remote runs, and VCS connections. We also offer paid plans for larger teams that include additional collaboration and governance features.

Each higher paid upgrade plan is a strict superset of any lower plans — for example, the Team & Governance plan includes all of the features the Team plan.

Source: <https://www.terraform.io/cloud-docs/overview>

upvoted 7 times

 **krusty93** 8 months, 2 weeks ago

What you posted is saying that the free tier doesn't support team management. And actually, it doesn't (Unified workflow tab): <https://www.hashicorp.com/products/terraform/pricing>

upvoted 2 times

 **master9** Most Recent 4 days, 15 hours ago

**Selected Answer: A**

All Terraform Cloud tiers (Free, Team, Business, and Enterprise) support team management and governance features. These features include roles and permissions, organization management, workspace access controls, and audit logging capabilities. They are designed to facilitate collaborative infrastructure management across teams of various sizes and organizational structures.

upvoted 1 times

 **ravi135** 5 months, 1 week ago

B is right

upvoted 1 times

 **itstammy** 6 months ago

**Selected Answer: B**

<https://spacelift.io/blog/terraform-cloud-pricing>

upvoted 1 times

 **samimshaikh** 6 months, 2 weeks ago

**Selected Answer: B**

<https://www.hashicorp.com/products/terraform/pricing>

upvoted 1 times

 **[Removed]** 7 months ago

**Selected Answer: B**

Cloud Free tier doesn't support team management.

upvoted 2 times

 **KingFvsher** 10 months ago

Option \*\*A. True\*\* is correct.

All tiers of Terraform Cloud, including the Free tier, support team management and governance. The Free tier includes everything needed to use Terraform in a team setting, including remote state, remote runs, private registry, secure variables, dynamic provider credentials, and additional security features for a robust security posture from the start. It also includes unlimited users and applies, so organizations can onboard any team member to collaborate, provision, and manage their own resources.

For larger teams with additional collaboration and governance features, there are paid plans. The Standard tier is great for organizations starting to use Terraform where team management, scalability, security, and support are critical. The Plus and Enterprise tiers offer even more advanced features for standardizing and managing infrastructure automation and lifecycle.

So yes, all Terraform Cloud tiers support team management and governance.

upvoted 2 times

 **zanhsieh** 1 year, 2 months ago

**Selected Answer: B**

"Cloud Free" does not provide "Team management". In below link, click "Unified workflow management" under "Features" and search the whole page (Ctrl-F) with "Team management".

<https://www.hashicorp.com/products/terraform/pricing>

upvoted 3 times

 **craiglee** 1 year, 2 months ago

It's true - All Terraform Cloud tiers, including the free tier, support team management and governance features. This includes the ability to manage access controls, view and audit changes made to infrastructure, and enforce policies across multiple workspaces. However, some features may be limited in the free tier, such as the number of users and workspaces that can be created, as well as the level of support provided by HashiCorp.

upvoted 2 times

 **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: A**

All Terraform Cloud tiers support team management and governance, including Free, Team, and Governance.

upvoted 1 times

 **macross** 1 year, 7 months ago

The keyword here is "All" Not true - so B.

upvoted 3 times

 **anand4nithi** 1 year, 7 months ago

**Selected Answer: B**

For Ex: Free tier only supports:

OPEN SOURCE FEATURES, PLUS:

State management

Remote operations

Private module registry

upvoted 2 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: B**

Free Tier does not support Team and Governance

<https://www.hashicorp.com/products/terraform/pricing>

upvoted 2 times

 **keiffo2** 1 year, 10 months ago

false - free tier doesn't support Governance

upvoted 4 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

Question #137

Topic 1

What advantage does an operations team that uses infrastructure as code have?

- A. The ability to delete infrastructure

- B. The ability to update existing infrastructure
- C. The ability to reuse best practice configurations and settings
- D. The ability to autoscale a group of servers

**Correct Answer: D**

*Community vote distribution*

C (100%)

 **bora4motion** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

C sounds good to me too.

upvoted 5 times

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

This should be it : "The ability to reuse best practice configurations and settings"

upvoted 5 times

 **keiffo2** 1 year, 10 months ago

C surely yes

upvoted 2 times

 **Bere** Most Recent 6 months ago

**Selected Answer: C**

While options B and D are also advantages associated with IaC, option C is the most directly related to the reuse of configurations and settings. Option A, the ability to delete infrastructure, is a basic function of any infrastructure management approach, not an advantage specific to IaC.

upvoted 2 times

 **cananmertesee** 9 months, 2 weeks ago

**Selected Answer: C**

This is the most direct benefit of IaC. With IaC, operations teams can version control their infrastructure configurations, ensuring that best practices are consistently applied across environments

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: C**

C. The ability to reuse best practice configurations and settings.

An operations team that uses infrastructure as code has the advantage of being able to reuse best practice configurations and settings across their infrastructure. This enables the team to quickly and easily provision new resources, and ensures that resources are configured consistently across the infrastructure. By defining infrastructure as code, the team can also automate the provisioning and configuration of resources, which helps reduce the risk of manual errors and frees up time for more important tasks.

upvoted 3 times

 **ruganesh** 1 year, 5 months ago

C is the right one

upvoted 2 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: C**

C is the answer

upvoted 2 times

**Original configuration:**

```
resource "aws_security_group" "http" {
 name = "http"
 ingress {
 from_port = "80"
 to_port = "80"
 protocol = "tcp"
 cidr_blocks = ["0.0.0.0/0"]
 }
}
```

**Updated configuration:**

```
resource "aws_security_group" "http" {
 name = "http"
 ingress {
 from_port = "80"
 to_port = "80"
 protocol = "tcp"
 cidr_blocks = ["0.0.0.0/0"]
 }
}
```

Which of the following commands would you run to update the ID in state without destroying the resource?

- A. terraform mv aws\_security\_group.http aws\_security\_group.http
- B. terraform apply
- C. terraform refresh

**Correct Answer: B**

*Community vote distribution*

A (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: A**

I guess it had to be terraform state mv .....

The terraform state mv command changes which resource address in your configuration is associated with a particular real-world object. Use to preserve an object when renaming a resource, or when moving a resource into or out of a child module.

upvoted 12 times

 **adouban** Highly Voted 1 year, 7 months ago

**Selected Answer: A**

A is correct but the command is wrong it should be terraform state mv <old\_name> <new\_name>

upvoted 8 times

 **ravi135** Most Recent 5 months, 1 week ago

A is correct

upvoted 1 times

 **samimshaikh** 6 months, 2 weeks ago

**Selected Answer: A**

options are wrong and it terraform state mv oldid newid

upvoted 1 times

 **bizimunda** 8 months ago

there is no such command as of 2023. The correct command is terraform state mv

upvoted 2 times

 **camps** 1 year, 3 months ago

**Selected Answer: A**

A. terraform mv aws\_security\_group.hpt aws\_security\_group.http.

The terraform mv command is used to rename a resource in your Terraform state without destroying it. In this case, you have fixed a typo in the Terraform ID of a resource, so you can use the terraform mv command to update the ID in the state without destroying the resource.

upvoted 2 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: A**

Terraform apply will delete the security group htp and create a new security group with the name http

upvoted 1 times

 **shopkitty** 1 year, 10 months ago

**Selected Answer: A**

if use the apply then it will destroy and re-create the resource, only terraform state mv allows to update the state file without destroying resources.

upvoted 3 times

You are creating a Terraform configuration which needs to make use of multiple providers, one for AWS and one for Datadog.

Which of the following provider blocks would allow you to do this?

A.

```
provider {
 "aws" {
 profile = var.aws_profile
 region = var.aws_region
 }

 "datadog" {
 api_key = var.datadog_api_key
 app_key = var.datadog_app_key
 }
}
```

B.

```
provider "aws" {
 profile = var.aws_profile
 region = var.aws_region
}

provider "datadog" {
 api_key = var.datadog_api_key
 app_key = var.datadog_app_key
}
```

C.

```
terraform {
 provider "aws" {
 profile = var.aws_profile
 region = var.aws_region
 }

 provider "datadog" {
 api_key = var.datadog_api_key
 app_key = var.datadog_app_key
 }
}
```

**Correct Answer: B**

✉  **keiffo2**  1 year, 10 months ago

I think B is correct as you are configuring the provider not declaring it

```
terraform {
 required_providers {
 aws = {
 source = "hashicorp/aws"
 version = "~> 4.0"
 }
 }
}
```

# Configure the AWS Provider  
provider "aws" {

```
 region = "us-east-1"
}
}
```

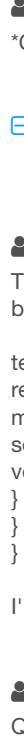
upvoted 13 times

✉  **depal\_dhir**  1 year, 10 months ago

B is correct

<https://www.terraform.io/language/providers/configuration>

upvoted 8 times

✉  **sjokkorein**  3 months ago

\*CREATING\* then it should be C as you need the root terraform block

upvoted 2 times

✉  **8876ca1** 3 weeks, 4 days ago

Provider block doesn't belong inside the terraform block :D

upvoted 1 times

✉  **alen995454** 6 months ago

The question seems to hinge on the word "needs" ... if it was asking for "required\_providers" then they would need to be nested inside a terraform block however none of the examples show the correct syntax for a required\_providers block

```
terraform {
 required_providers {
 mycloud = {
 source = "mycorp/mycloud"
 version = "~> 1.0"
 }
 }
}
```

I'm going with B.

upvoted 1 times

✉  **Spandrop** 7 months ago

Question says "you are creating" your configuration, not configuring. It should be C

upvoted 1 times

✉  **Spandrop** 7 months ago

well paying more attention to the alternatives, I think none is correct

upvoted 1 times

✉  **Tyler2023** 8 months, 2 weeks ago

<https://developer.hashicorp.com/terraform/language/providers/requirements>

upvoted 1 times

Question #140

*Topic 1*

Terraform variable names are saved in the state file.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **keiffo2** Highly Voted 1 year, 10 months ago

terraform state is a representation of your infrastructure, not of your config file, so variables wouldn't be stored in state. FALSE  
upvoted 8 times

 **camps** Highly Voted 1 year, 3 months ago

**Selected Answer: B**

B. False.

Terraform variable names are not saved in the state file. The state file contains information about the resources that Terraform manages, such as their current state and metadata. Variables are used in your Terraform configuration to parameterize your code and make it more reusable, but they are not saved in the state file. Instead, variable values are provided at runtime, either by passing them on the command line, through environment variables, or by using default values defined in your configuration.

upvoted 6 times

 **Nunyabiznes** Most Recent 1 year, 3 months ago

**Selected Answer: B**

B. False. Terraform variable names are not saved in the state file. The state file contains the actual values of the resources that were created or modified by Terraform.

upvoted 1 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: B**

I checked in my lab. variable name is not stored in state file

upvoted 3 times

 **dani88ge** 1 year, 9 months ago

**Selected Answer: B**

Question #141

Topic 1

Terraform Cloud is available only as a paid offering from HashiCorp.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Burakko** Highly Voted  1 year, 10 months ago

**Selected Answer: B**

"Terraform Cloud is a commercial SaaS product developed by HashiCorp. Many of its features are free for small teams, including remote state storage, remote runs, and VCS connections. We also offer paid plans for larger teams that include additional collaboration and governance features."

upvoted 7 times

 **Nunyabiznes** Most Recent  1 year, 3 months ago

**Selected Answer: B**

B. False. Terraform variable names are not saved in the state file. The state file contains the actual values of the resources that were created or modified by Terraform.

upvoted 1 times

 **odisor** 1 year, 7 months ago

is free up to 5 active users

upvoted 3 times

 **RVivek** 1 year, 9 months ago

Free tier is available, however team and governance features are not available

upvoted 1 times

 **Nunyabiznes** 1 year, 3 months ago

Not exactly, the free version of Terraform Cloud includes Terraform teams and basic governance features, such as role-based access control (RBAC), policy as code with Sentinel, and audit logs. However, more advanced features like SSO and custom policy checks require a paid subscription.

upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

Many of Terraform Cloud features are free for small teams, including remote state storage, remote runs, and VCS connections.

upvoted 2 times

 **keiffo2** 1 year, 10 months ago

terraform cloud has a free option - false.

Paid for options are "team and governance" and "Business"

upvoted 1 times

Question #142

Topic 1

Which of the following is not a way to trigger terraform destroy?

- A. Using the destroy command with auto-approve
- B. Running terraform destroy from the correct directory and then typing "yes" when prompted in the CLI
- C. Passing --destroy at the end of a plan request

D. Delete the state file and run terraform apply

**Correct Answer: D**

*Community vote distribution*

C (51%)

D (49%)

 **lizzard812** Highly Voted 1 year, 3 months ago

The real exam gives options either apply -destroy or delete state file + plan, so obviously plan does not destroy anything, so I went with this. 1 examtopics question here is misspelled. So you won't be confused that much on real exam.

upvoted 16 times

 **wanrltw** Highly Voted 1 year, 7 months ago

I'd go with D.

The `terraform plan -destroy` command will only show what is going to be destroyed (it's only a plan).

However, removing the state file has nothing to do with `terraform destroy` at all - it would only make Terraform forget about its objects while it continues to exist in the remote system.

upvoted 13 times

 **Daminaij** Most Recent 1 week, 4 days ago

Plan don't destroy anything

upvoted 1 times

 **dzhang344** 3 weeks, 2 days ago

**Selected Answer: C**

execute plan doesn't trigger destroy

upvoted 1 times

 **Mouszie** 1 month, 2 weeks ago

will go for D

upvoted 1 times

 **Edward2021** 1 month, 2 weeks ago

"is not a way to do " - ill take A.. you shouldn't do destroy without being sure what you are destroying..

upvoted 1 times

 **SilentH** 2 months, 4 weeks ago

**Selected Answer: C**

Because there is no double-dash destroy (hoping this isn't a typo)

upvoted 1 times

 **Ryan1002** 3 months, 3 weeks ago

**Selected Answer: D**

Destroy mode: creates a plan whose goal is to destroy all remote objects that currently exist, leaving an empty Terraform state. It is the same as running terraform destroy. Destroy mode can be useful for situations like transient development environments, where the managed objects cease to be useful once the development task is complete.

Activate destroy mode using the -destroy command line option.

upvoted 1 times

 **recep38** 4 months, 3 weeks ago

I tested in a lab. When you run "terraform plan -destroy" it just shows which resource will be destroyed but didn't destroy anything. If you want to destroy you should run "terraform destroy". But interestingly when you delete state file and run "terraform apply" command it creates a new resources not destroy. It seems two options are correct. C and D.

upvoted 2 times

 **vipulchoubisa** 6 months, 1 week ago

delete state file is never suggested by terraform so simple, answer is D.

upvoted 1 times

✉️  **samimshaikh** 6 months, 2 weeks ago

**Selected Answer: C**

Answer is C

because there is no such command to trigger a destroy "terraform destroy --destroy"

A & B are way to trigger a destroy.

D: will not suite because in the question it has mentioned that "way of triggering a destroy" instead of "way of destroy"  
upvoted 1 times

✉️  **March2023** 1 year, 1 month ago

**Selected Answer: D**

im going with D

upvoted 1 times

✉️  **sdm13168** 1 year, 1 month ago

**Selected Answer: D**

D, if you delete the state file, then run terraform apply, terraform will try to create the whole infrastructure but get error.  
upvoted 1 times

✉️  **FawadK** 1 year, 2 months ago

**Selected Answer: D**

D makes more sense than C. Correct answer should be D

upvoted 1 times

✉️  **zanhsieh** 1 year, 2 months ago

**Selected Answer: D**

D.

C: Wrong. "Destroy mode: creates a plan whose goal is to destroy all remote objects that currently exist, leaving an empty Terraform state. It is the same as running terraform destroy."

<https://developer.hashicorp.com/terraform/cli/commands/plan#planning-modes>

upvoted 2 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer: C**

C. Passing --destroy at the end of a plan request.

The terraform destroy command is used to destroy the infrastructure resources created by Terraform according to the current configuration.

Option D is incorrect because deleting the state file and running terraform apply will not trigger terraform destroy. Running terraform apply without changes to the Terraform configuration will have no effect on the existing infrastructure resources, and will not destroy them. If you delete the state file and then run terraform apply, Terraform will not know about any existing infrastructure resources, and will attempt to create new resources according to the current configuration. This may lead to unexpected results and is not a recommended approach to managing infrastructure with Terraform.

upvoted 2 times

✉️  **SilentMilli** 1 year, 3 months ago

**Selected Answer: D**

terraform apply is a command that creates or updates resources as described in the Terraform configuration. Deleting the state file and running terraform apply will cause Terraform to create resources that are missing, rather than destroying them.

upvoted 1 times

Which of the following is not an advantage of using infrastructure as code operations?

- A. Self-service infrastructure deployment
- B. Troubleshoot via a Linux diff command

- C. Public cloud console configuration workflows
- D. Modify a count parameter to scale resources
- E. API driven workflows

**Correct Answer: B**

*Community vote distribution*

C (78%)

B (22%)

 **camps** Highly Voted 1 year, 3 months ago

**Selected Answer: C**

C. Public cloud console configuration workflows.

Public cloud console configuration workflows are not an advantage of using infrastructure as code operations, but are instead a separate method of managing cloud infrastructure. Infrastructure as code is focused on defining and managing infrastructure resources through code, using tools like Terraform or CloudFormation.

Option A, self-service infrastructure deployment, allows developers and operations teams to provision resources on-demand, without having to request manual intervention from infrastructure teams.

Option B, troubleshoot via a Linux diff command, allows teams to identify changes made to infrastructure configurations over time, and compare the differences between different versions of the code.

Option D, modify a count parameter to scale resources, allows teams to easily scale resources up or down, without having to manually provision or deprovision resources.

Option E, API driven workflows, allows teams to integrate infrastructure management with other automation tools and processes, and build custom workflows to meet their specific needs.

upvoted 10 times

 **Nunyabiznes** 1 year, 3 months ago

I smell chat-GPT

upvoted 15 times

 **gold4otas** 6 months, 2 weeks ago

Exactly! ChatGPT for sure.

upvoted 1 times

 **NashP** Most Recent 5 months, 1 week ago

C. Public cloud console configuration workflows: Using console-based configuration in a public cloud does not align with the principles of IaC. IaC promotes defining infrastructure through code rather than using graphical interfaces.

upvoted 1 times

 **David\_C\_90** 1 year, 4 months ago

**Selected Answer: C**

Answer is C

Regarding B, since infrastructure is defined as code (files) we can use the diff command to check for differences

upvoted 4 times

 **Pietjeplukgeluk** 1 year, 5 months ago

C looks to me most correct answer. B is strange to, not sure why this is not also wrong, but C looks best anyway.

upvoted 4 times

 **jackn** 1 year, 5 months ago

it is C

Public cloud console configuration workflows - typically involve manual steps therefore nothing to do with IaC , certainly not being an advantage of IaC

upvoted 4 times

✉  **lezgino** 1 year, 5 months ago

Selected Answer: C

Advantages of using infrastructure as code operations include:

Self-service infrastructure deployment

Troubleshooting via a Linux diff command

API driven workflows

Modifying a count parameter to scale resources

The use of a public cloud console configuration workflow is not an advantage of using infrastructure as code operations.

upvoted 3 times

✉  **dепал\_dхир** 1 year, 10 months ago

**Selected Answer: B**

This is a Linux shell command and has nothing to do with IaC

upvoted 3 times

✉  **tycho** 1 year, 2 months ago

it has everything to do with IaC, because you cannot diff the management console UI, but you can easily diff the terraform files, therefore a advantage to use the IaC

upvoted 1 times

✉  **Optimus** 1 year, 10 months ago

**Selected Answer: B**

B is correct since terraform is used to deploy the infrastructure, not to troubleshoot it

upvoted 1 times

Question #144

Topic 1

You're writing a Terraform configuration that needs to read input from a local file called id\_rsa.pub.

Which built-in Terraform function can you use to import the file's contents as a string?

- A. fileset("id\_rsa.pub")
- B. filebase64("id\_rsa.pub")
- C. templatefile("id\_rsa.pub")
- D. file("id\_rsa.pub")

**Correct Answer: A**

*Community vote distribution*

D (100%)

✉  **Uma10** Highly Voted 1 year, 10 months ago

**Selected Answer: D**

file reads the contents of a file at the given path and returns them as a string.

Source: <https://www.terraform.io/language/functions/file>

upvoted 8 times

✉  **ravi135** Most Recent 5 months, 1 week ago

D is right

upvoted 1 times

✉️  **NashP** 5 months, 1 week ago

D. file("id\_rsa.pub")

Explanation: To read the contents of a local file as a string in Terraform, you can use the file function. The correct syntax is:

```
variable "public_key" {
```

```
 type = string
```

```
 default = file("id_rsa.pub")
```

```
}
```

This will read the contents of the "id\_rsa.pub" file and assign it to the variable "public\_key" as a string.

upvoted 2 times

✉️  **Bere** 6 months ago

**Selected Answer: D**

Example:

```
resource "aws_key_pair" "example" {
 key_name = "my-key-pair"
 public_key = file("${path.module}/id_rsa.pub")
}
```

upvoted 1 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer: D**

D. file("id\_rsa.pub").

The file function is a built-in Terraform function that can be used to read the contents of a local file and return the contents as a string. The file function takes a single argument, which is the path to the file to read.

upvoted 1 times

✉️  **RVivek** 1 year, 9 months ago

**Selected Answer: D**

<https://www.terraform.io/language/functions/file>  
<https://www.terraform.io/language/functions/fileset>

upvoted 1 times

✉️  **Pinky0289** 1 year, 10 months ago

file ("file\_path") reads the contents of the file and returns the string, so option D is correct.

upvoted 1 times

✉️  **far12** 1 year, 10 months ago

**Selected Answer: D**

the correct answer is D , explain :

- file reads the contents of a file at the given path and returns them as a string.

- filebase64 also reads the contents of a given file, but returns the raw bytes in that file Base64-encoded.

-templatefile renders using a file from disk as a template.

upvoted 2 times

✉️  **shopkitty** 1 year, 10 months ago

**Selected Answer: D**

fileset returning file name with full path, file will read the content of the files.

upvoted 1 times

✉️  **keiffo2** 1 year, 10 months ago

<https://www.terraform.io/language/functions/file>

Answer D

upvoted 1 times

✉️  **Burakko** 1 year, 10 months ago

**Selected Answer: D**

"file reads the contents of a file at the given path and returns them as a string."

file(path)

upvoted 3 times

What does Terraform use providers for? (Choose three.)

- A. Provision resources for on-premises infrastructure services
- B. Simplify API interactions
- C. Provision resources for public cloud infrastructure services
- D. Enforce security and compliance policies
- E. Group a collection of Terraform configuration files that map to a single state file

**Correct Answer:** ABC

*Community vote distribution*

|           |           |    |
|-----------|-----------|----|
| ABC (68%) | BCE (26%) | 6% |
|-----------|-----------|----|

 **waldonuts** Highly Voted 1 year, 10 months ago

**Selected Answer:** ABC

A and C are the same with a alternate target, and terraform can provision to local on prem resources (VMware for example) , B is pretty much reason providers are there for me.

upvoted 14 times

 **bora4motion** Highly Voted 1 year, 10 months ago

**Selected Answer:** BCE

to me BCE sounds ok

upvoted 6 times

 **[Removed]** Most Recent 7 months ago

**Selected Answer:** ABC

A,B,C sounds good to me.

upvoted 1 times

 **kiran15789** 1 year, 2 months ago

**Selected Answer:** ABC

Terraform can be used for Provision resources for on-premises infrastructure services

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer:** BCE

B. Simplify API interactions.

C. Provision resources for public cloud infrastructure services.

E. Group a collection of Terraform configuration files that map to a single state file.

Terraform uses providers to interact with infrastructure services, both on-premises and in the public cloud. Providers are responsible for managing the lifecycle of resources, including creating, updating, and deleting resources as needed. Providers also simplify API interactions by abstracting away the details of the underlying APIs and presenting a unified interface to Terraform.

Enforcing security and compliance policies is not a primary responsibility of providers, although some providers may offer features to help with these tasks.

Option A is incorrect because while providers can be used to manage on-premises infrastructure services, this is not their only purpose.

Option E is incorrect because while Terraform workspaces can be used to group a collection of Terraform configuration files that map to a single state file, this is not the purpose of providers.

upvoted 2 times

 **sagunala5** 1 year, 3 months ago

ABC

<https://developer.hashicorp.com/terraform/language/providers#what-providers-do>

upvoted 1 times

 **kennynelcon** 1 year, 5 months ago

**Selected Answer: ABC**

Def not D and E

upvoted 1 times

 **Zeppoonstream** 1 year, 5 months ago

A. Provision resources for on-premises infrastructure services: Providers can be used to provision resources for on-premises infrastructure services like VMWare, OpenStack, etc.

C. Provision resources for public cloud infrastructure services: Providers can be used to provision resources for public cloud infrastructure services like AWS, GCP, Azure, etc.

D. Enforce security and compliance policies: Providers can be used to enforce security and compliance policies like IAM roles and policies, security groups, firewall rules, etc.

upvoted 3 times

 **lezgino** 1 year, 5 months ago

correct

upvoted 1 times

 **robertninho** 1 year, 6 months ago

BCD, Option A (provision resources for on-premises infrastructure services) is not necessarily true, as not all providers support on-premises infrastructure. Option E (group a collection of Terraform configuration files that map to a single state file) is not related to providers.

upvoted 1 times

 **robertninho** 1 year, 6 months ago

My bad, ABC

upvoted 2 times

 **adouban** 1 year, 7 months ago

**Selected Answer: ABC**

ABC for me

upvoted 1 times

 **itsVespucci** 1 year, 8 months ago

ABC 4 sure

upvoted 1 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: ABC**

Provider is used to provision resources A on prem C on cloud. B to use API communication

upvoted 3 times

 **babuappnet** 1 year, 9 months ago

sure for BCE

upvoted 1 times

 **Fati\_2022** 1 year, 10 months ago

ABC

A Terraform Provider represents an integration that is responsible for understanding API interactions with the underlying infrastructure, such as a public cloud service (AWS, GCP, Azure), a PaaS service (Heroku), a SaaS service (DNSimple, CloudFlare), or on-prem resources (vSphere).

upvoted 4 times

 **keiffo2** 1 year, 10 months ago

ABC - for me too

upvoted 4 times

 **donathon** 1 year, 10 months ago

**Selected Answer: BCD**

E is talking about module. "Terraform evaluates all of the configuration files in a module, effectively treating the entire module as a single document."

upvoted 2 times

You can reference a resource created with `for_each` using a Splat (\*) expression.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (73%)

A (27%)

 **Nunyabiznes** Highly Voted 1 year, 3 months ago

**Selected Answer: A**

```
resource "aws_instance" "example" {
 for_each = {
 "web-1" = "ami-0c55b159cbfabe1f0"
 "web-2" = "ami-0c55b159cbfabe1f0"
 }

 instance_type = "t2.micro"
 ami = each.value
}

Accessing a single instance
output "web_1_id" {
 value = aws_instance.example["web-1"].id
}

Accessing all instances using a Splat expression
output "instance_ids" {
 value = aws_instance.example.*.id
}
```

In this example, `aws_instance.example` creates two EC2 instances using the same AMI, and `for_each` is used to create the instances with distinct identifiers. The first output references a single instance by its identifier, while the second output uses a Splat expression (`aws_instance.example[*]`) to reference all instances and return a list of their IDs.

upvoted 7 times

 **chxnedu** Most Recent 3 months, 4 weeks ago

B

"Resources that use the `for_each` argument will appear in expressions as a map of objects, so you can't use splat expressions with those resources."

<https://developer.hashicorp.com/terraform/language/expressions/splat>

upvoted 3 times

 **Kaname93** 4 months, 2 weeks ago

**Selected Answer: B**

From the documentation :

Resources that use the `for_each` argument will appear in expressions as a map of objects, so you can't use splat expressions with those resources. For more information, see Referring to Resource Instances.

upvoted 3 times

✉  **NashP** 5 months, 1 week ago

When using the `for_each` argument to create multiple instances of a resource in Terraform, you can reference the attributes of those instances using a Splat expression (\*). The Splat expression is used to extract values from a collection of resources.

```
variable "instance_names" {
 type = set(string)
 default = ["web-1", "web-2", "web-3"]
}

resource "aws_instance" "example" {
 for_each = var.instance_names

 ami = "ami-0123456789abcdef0"
 instance_type = "t2.micro"
 key_name = "your-key-pair-name"
 tags = {
 Name = each.key
 }
}

Reference the attributes using Splat expression
output "instance_ids" {
 value = aws_instance.example[*].id
}
```

A:TURE

upvoted 2 times

✉  **TigerInTheCloud** 6 months, 4 weeks ago

**Selected Answer: B**

\$ terraform validate

```
| Error: Missing resource instance key
| on main.tf line 15, in output "ec2s":
| 15: value = aws_instance.example.ebs_block_device.*.id
|
| Because aws_instance.example has "for_each" set, its attributes must be accessed on specific instances.
|
| For example, to correlate with indices of a referring resource, use:
| aws_instance.example[each.key]
```

upvoted 1 times

✉  **BaburTurk** 10 months, 3 weeks ago

**Selected Answer: B**

You cannot reference a resource created with `for_each` using a splat (\*) expression. Resources that use the `for_each` argument will appear in expressions as a map of objects, so you can't use splat expressions and you have to use a for expression loop.

For example, the following code will create two EC2 instances:

```
resource "aws_instance" "web" {
 for_each = [1, 2]
```

```
 ami = "ami-0123456789abcdef0"
 instance_type = "t2.micro"
}
```

You cannot reference the EC2 instances created by this code using a splat expression, such as `*aws_instance.web`. Instead, you would have to use a for expression loop, such as:

```
for i, instance in aws_instance.web.items() {
 # Do something with the instance
}
```

upvoted 3 times

✉  **kiran15789** 1 year, 2 months ago

```
aws_instance.example[*].id
this is valid
```

upvoted 1 times

✉  **gspb** 1 year, 2 months ago

**Selected Answer: B**

Option is the correct answer.

"You can't refer to a resource using the splat operator [\*] if the resource uses a for\_each argument. This is because in this case the resource is map of objects rather than a list of objects. The splat operator applies only to lists."

<https://www.terraformbyexample.com/splat/>

upvoted 4 times

✉  **gspb** 1 year, 2 months ago

\*option B is the correct answer

upvoted 1 times

✉  **FarziWaliMarzi** 1 year, 2 months ago

**Selected Answer: B**

<https://developer.hashicorp.com/terraform/language/expressions/splat#legacy-attribute-only-splat-expressions>

A splat expression provides a more concise way to express a common operation that could otherwise be performed with a for expression.

Hence B

upvoted 1 times

✉  **Fliyd** 1 year, 3 months ago

**Selected Answer: A**

There is a difference between splat and LEGACY splat. in this case, a legacy splat can be used, even if depreciated.

source : <https://developer.hashicorp.com/terraform/language/expressions/splat#legacy-attribute-only-splat-expressions>

upvoted 1 times

✉  **David\_C\_90** 1 year, 4 months ago

**Selected Answer: B**

The splat expression patterns shown above apply only to lists, sets, and tuples. To get a similar result with a map or object value you must use for expressions.

Resources that use the for\_each argument will appear in expressions as a map of objects, so you can't use splat expressions with those resources. For more information, see Referring to Resource Instances.

<https://developer.hashicorp.com/terraform/language/expressions/splat#splat-expressions-with-maps>

upvoted 1 times

✉  **Daro\_** 1 year, 5 months ago

**Selected Answer: B**

B

<https://www.terraformbyexample.com/splat/>

You can't refer to a resource using the splat operator [\*] if the resource uses a for\_each argument. This is because in this case the resource is map of objects rather than a list of objects. The splat operator applies only to lists.

upvoted 3 times

✉  **leznino** 1 year, 5 months ago

Selected Answer: A

You can reference a resource created with for\_each using a Splat () expression, also known as the "splat operator". The Splat operator allows to reference all of the instances of a resource created with the for\_each argument in a single reference, by using an asterisk () in the reference. This can be useful when you need to reference all instances of a resource in a single expression, such as when setting up dependencies between resources or referencing outputs from multiple resources.

upvoted 1 times

✉  **Abuu** 1 year, 5 months ago

**Selected Answer: A**

A Splat (\*) expression allows you to reference a resource created with for\_each by expanding the expression into a list of individual resource references. This makes it easier to access the resources, as you don't have to reference each one individually.

upvoted 1 times

 **dinesh198728** 1 year, 10 months ago

**Selected Answer: B**

Splat Expressions with Maps

The splat expression patterns shown above apply only to lists, sets, and tuples. To get a similar result with a map or object value you must use for expressions.

Resources that use the for\_each argument will appear in expressions as a map of objects, so you can't use splat expressions with those resources. For more information, see Referring to Resource Instances.

[https://www.terraform.io/language/meta-arguments/for\\_each#referring-to-instances](https://www.terraform.io/language/meta-arguments/for_each#referring-to-instances)

upvoted 4 times

 **[Removed]** 1 year, 10 months ago

**Selected Answer: B**

Note that unlike count, splat expressions are not directly applicable to resources managed with for\_each, as splat expressions must act on a list value. However, you can use the values() function to extract the instances as a list and use that list value in a splat expression:

`values(aws_instance.example)[*].id`

<https://www.terraform.io/language/expressions/references>

upvoted 3 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

<https://www.terraform.io/language/expressions/splat>

upvoted 1 times

Question #147

*Topic 1*

How does Terraform determine dependencies between resources?

- A. Terraform automatically builds a resource graph based on resources, provisioners, special meta-parameters, and the state file, if present.
- B. Terraform requires all dependencies between resources to be specified using the depends\_on parameter
- C. Terraform requires resources in a configuration to be listed in the order they will be created to determine dependencies
- D. Terraform requires resource dependencies to be defined as modules and sourced in order

**Correct Answer: A**

*Community vote distribution*

A (100%)

👤 **depal\_dhir** Highly Voted 1 year, 10 months ago

**Selected Answer: A**

<https://learn.hashicorp.com/tutorials/terraform/dependencies>

upvoted 5 times

👤 **Bere** Most Recent 6 months ago

**Selected Answer: A**

<https://developer.hashicorp.com/terraform/tutorials/configuration-language/dependencies>

Most of the time, Terraform infers dependencies between resources based on the configuration given, so that resources are created and destroyed in the correct order. Occasionally, however, Terraform cannot infer dependencies between different parts of your infrastructure, and will need to create an explicit dependency with the depends\_on argument.

upvoted 2 times

👤 **SilentMilli** 1 year, 3 months ago

**Selected Answer: A**

When Terraform evaluates a configuration, it automatically creates a graph of all the resources and their dependencies. Terraform uses this graph to determine the order in which resources need to be created, updated, or destroyed. Terraform looks at resource dependencies based on implicit and explicit relationships between resources, which are defined using properties like count, for\_each, depends\_on, and others. Terraform can also use the state file to track dependencies between resources and maintain the order of operations during updates

upvoted 2 times

👤 **chael88** 1 year, 5 months ago

Took the exam and passed. There was a similar question like this not listed here on ExamTopics. It goes like this:

How does Terraform handle dependencies?

- A. By using depends\_on
- B. Terraform will automatically handle some resource dependencies

I cant remember the other options, but I selected B.

upvoted 1 times

👤 **keiffo2** 1 year, 10 months ago

A is the correct answer - by way of eliminating the other answers

upvoted 3 times

Which parameters does terraform import require? (Choose two.)

- A. Path
- B. Provider
- C. Resource ID
- D. Resource address

**Correct Answer:** BC

*Community vote distribution*

CD (100%)

 dc\_98 Highly Voted 1 year, 10 months ago

**Selected Answer:** CD

its CD shown here: <https://www.terraform.io/cli/commands/import#usage>  
upvoted 9 times

 Jlee7 Most Recent 1 year ago

C&D for SURE!  
upvoted 1 times

 RVivek 1 year, 9 months ago

**Selected Answer:** CD

the command is terraform import <resource address> <resource id>  
upvoted 4 times

 shopkitty 1 year, 10 months ago

**Selected Answer:** CD

should be C & D  
upvoted 2 times

 bora4motion 1 year, 10 months ago

**Selected Answer:** CD

this one's easy CD.  
upvoted 4 times

Once a new Terraform backend is configured with a Terraform code block, which command(s) is (are) used to migrate the state file?

- A. terraform apply
- B. terraform push
- C. terraform destroy, then terraform apply
- D. terraform init

**Correct Answer:** D

*Community vote distribution*

D (76%)

B (24%)

✉  **keiffo2** Highly Voted 1 year, 10 months ago

D: I think

Migrate the state file

Once you have authenticated to Terraform Cloud, you're ready to migrate your local state file to Terraform Cloud. To begin the migration, reinitialize. This causes Terraform to recognize your cloud block configuration.

\$ `terraform init`

Initializing Terraform Cloud...

Do you wish to proceed?

As part of migrating to Terraform Cloud, Terraform can optionally copy your current workspace state to the configured Terraform Cloud workspace.

Answer "yes" to copy the latest state snapshot to the configured Terraform Cloud workspace.

Answer "no" to ignore the existing state and just activate the configured Terraform Cloud workspace with its existing state, if any.

Should Terraform migrate your existing state?

Enter a value:

Copy

During reinitialization, Terraform presents a prompt saying that it will copy the state file to your Terraform Cloud workspace. Enter yes and Terraform will migrate the state from your local machine to Terraform Cloud.

upvoted 23 times

✉  **nakuadaml** Most Recent 4 months ago

Selected Answer: D

<https://developer.hashicorp.com/terraform/tutorials/cloud/cloud-migrate>

Important: The `terraform push` command is no longer functional.

<https://developer.hashicorp.com/terraform/cli/commands/push>

upvoted 1 times

✉  **[Removed]** 7 months ago

Selected Answer: D

B is correct if the command is "`terraform state push`"

upvoted 1 times

✉  **DevoteamAnalytix** 1 year, 2 months ago

I think it's B not D because it is about MIGRATING the state file.

"Usage: `terraform state push [options] PATH`"

<https://developer.hashicorp.com/terraform/cli/commands/state/push>

"The `terraform push` command was an early implementation of remote Terraform runs. It allowed teams to push a configuration to a remote environment in a discontinued version of Terraform Enterprise."

<https://developer.hashicorp.com/terraform/cli/commands/push>

upvoted 2 times

✉  **Stanislav4907** 1 year, 3 months ago

Selected Answer: D

Once a new Terraform backend is configured with a Terraform code block, you can use the following command to migrate the state file from the previous backend to the new one:

`terraform init -migrate-state`

upvoted 3 times

✉  **Nunyabiznes** 1 year, 3 months ago

Selected Answer: D

Terraform push has been retired after version 0.12.0

upvoted 3 times

 **camps** 1 year, 3 months ago

**Selected Answer: D**

D. terraform init.

When configuring a new Terraform backend, the state file needs to be migrated to the new backend so that it can be used to manage infrastructure state going forward. The terraform init command is used to initialize a new backend and migrate the state file to that backend.

upvoted 2 times

 **mamoon\_malta2022** 1 year, 4 months ago

Answer D: The terraform state push command is used to manually upload a local state file to remote state. This command also works with local state.

This command should rarely be used. It is meant only as a utility in case manual intervention is necessary with the remote state.

upvoted 1 times

 **Americanman** 1 year, 4 months ago

The terraform state push command is used to manually upload a local state file to remote state. Would go with D

upvoted 2 times

 **z466235244** 1 year, 4 months ago

**Selected Answer: D**

Terraform push is no longer functional

upvoted 3 times

 **princajen** 1 year, 5 months ago

After configuring a new Terraform backend with a Terraform code block, you will need to use the terraform init command to initialize the backend and download any necessary provider plugins.

Once you have initialized the backend, you can use the terraform state command to manage the state file. If you have an existing state file that you need to migrate to the new backend, you can use the terraform state pull command to download the current state file, and then use the terraform state push command to upload it to the new backend.

upvoted 1 times

 **lezgino** 1 year, 5 months ago

The correct command to migrate the state file to a new backend is: "terraform init". The "terraform init" command will detect the new backend configuration and prompt you to copy your existing state to the new backend. B is correct

upvoted 1 times

 **Only5** 1 year, 5 months ago

This is from chatGPT - answer : B

The command used to migrate the state file to a new Terraform backend is terraform state push. This command uploads the local state file to the new backend, replacing the state that may already exist in the backend.

upvoted 1 times

 **ssanjayt** 1 year, 6 months ago

**Selected Answer: D**

Answer is D coz terraform push is deprecated and does not operate in the current version of TFE.

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

id go with B; terraform state push

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

I change my answer to D since you first have to terraform init and then use terraform state push. (Answer B says just terraform push, and that is not correct)

upvoted 3 times

 **alirasouli** 1 year, 7 months ago

**Selected Answer: D**

When you change a backend's configuration, you must run `terraform init` again to validate and configure the backend before you can perform any plans, applies, or state operations.

Reference:

<https://developer.hashicorp.com/terraform/language/settings/backends/configuration#initialization>

upvoted 3 times

 **ArizonaClassics** 1 year, 7 months ago

D: Command: state push

The terraform state push command is used to manually upload a local state file to remote state. This command also works with local state.

This command should rarely be used. It is meant only as a utility in case manual intervention is necessary with the remote state.  
upvoted 1 times

Question #150

Topic 1

What does this code do?

```
terraform {
 required_providers {
 aws = "~> 3.0"
 }
}
```

- A. Requires any version of the AWS provider >= 3.0 and < 4.0
- B. Requires any version of the AWS provider >= 3.0
- C. Requires any version of the AWS provider after the 3.0 major release, like 4.1
- D. Requires any version of the AWS provider > 3.0

**Correct Answer: A**

*Community vote distribution*

A (86%)

14%

 **DerekKey** Highly Voted 1 year, 7 months ago

A - ~>: Allows only the rightmost version component to increment. For example, to allow new patch releases within a specific minor release, use the full version number: ~> 1.0.4 will allow installation of 1.0.5 and 1.0.10 but not 1.1.0. This is usually called the pessimistic constraint operator  
<https://developer.hashicorp.com/terraform/language/expressions/version-constraints#version-constraint-syntax>

upvoted 13 times

 **mawish** Highly Voted 1 year, 9 months ago

**Selected Answer: A**

The ~> operator is a convenient shorthand for allowing the rightmost component of a version to increment.  
reference : <https://www.terraform.io/language/providers/requirements>

upvoted 6 times

 **mawish** 1 year, 9 months ago

Allows only the rightmost version component to increment. For example, to allow new patch releases within a specific minor release, use the full version number: ~> 1.0.4 will allow installation of 1.0.5 and 1.0.10 but not 1.1.0.

upvoted 2 times

 **NashP** Most Recent 5 months, 1 week ago

A. Requires any version of the AWS provider  $\geq 3.0$  and  $< 4.0$ .

Explanation:

The version constraint " $\sim >$ " specifies that the required version of the AWS provider should be at least 3.0 but less than 4.0. The ' $\sim >$ ' operator is used for optimistic version constraints and allows any version greater than or equal to the specified minimum version but less than the next major release.

So, the correct interpretation is option A. It requires any version of the AWS provider  $\geq 3.0$  and  $< 4.0$ .

upvoted 1 times

 **enook** 6 months, 2 weeks ago

Selected Answer: A

Correct answer is A

upvoted 1 times

 **[Removed]** 7 months ago

Selected Answer: A

The version must be less than 4.0

upvoted 1 times

 **Stanislav4907** 1 year, 3 months ago

Selected Answer: A

The  $\sim >$  syntax in the version constraint specifies that the AWS provider version must be greater than or equal to version 3.0, but less than version 4.0. This means that Terraform will only work with the AWS provider version 3.x, but will not work with version 4.x or any other major version.

upvoted 4 times

 **Nunyabiznes** 1 year, 3 months ago

Selected Answer: A

The clue here is " $\sim$ " ["pessimistic version constraint"], that is why it is A, but if it were " $\geq 3.0$ " then the answer would be B.

upvoted 2 times

 **camps** 1 year, 3 months ago

Selected Answer: A

A. Requires any version of the AWS provider  $\geq 3.0$  and  $< 4.0$ .

In Terraform, provider version constraints can be specified using version ranges. The version constraint  $\sim >$  3.0 specifies that the configuration requires any version of the AWS provider that is greater than or equal to version 3.0, but less than version 4.0.

upvoted 3 times

 **zecch** 1 year, 4 months ago

Selected Answer: A

definitely A.

upvoted 3 times

 **r1ck** 1 year, 4 months ago

atmost version - isn't it D ?

upvoted 1 times

 **mamoon\_malta2022** 1 year, 4 months ago

A is the correct Answer

$\sim >$ : Allows only the rightmost version component to increment. For example, to allow new patch releases within a specific minor release, use a full version number:  $\sim > 1.0.4$  will allow installation of 1.0.5 and 1.0.10 but not 1.1.0. This is usually called the pessimistic constraint operator.  
<https://developer.hashicorp.com/terraform/language/v1.2.x/expressions/version-constraints>

upvoted 2 times

 **kennynelcon** 1 year, 5 months ago

Selected Answer: A

Tilde sign is for last decimal

upvoted 2 times

 **Only5** 1 year, 5 months ago

Answer is A

upvoted 1 times

 **G4Exams** 1 year, 8 months ago

Selected Answer: A

If there not just 3 mentioned but 3.0 or 3.2 for example dann it includes all versions smaller then 4. So the right answer is for sure A.  
upvoted 2 times

 **G4Exams** 1 year, 8 months ago

If there not just 3 mentioned but 3.0 or 3.2 for example dann it includes all versions smaller then 4. So the right answer is for sure A.  
upvoted 1 times

 **Anderson01** 1 year, 8 months ago

I will go with A  
upvoted 1 times

 **alifie** 1 year, 9 months ago

Selected Answer: A

A is correct

upvoted 2 times

What does terraform refresh modify?

- A. Your cloud infrastructure
- B. Your state file
- C. Your Terraform plan
- D. Your Terraform configuration

**Correct Answer:** B

*Community vote distribution*

B (100%)

 **Uma10** Highly Voted 1 year, 10 months ago

The answer is correct.

The terraform refresh command reads the current settings from all managed remote objects and updates the Terraform state to match.

Source: <https://www.terraform.io/cli/commands/refresh>  
upvoted 5 times

 **camps** Most Recent 1 year, 3 months ago

**Selected Answer: B**

B. Your state file.

The terraform refresh command in Terraform is used to update the Terraform state file with the current real-world state of the infrastructure. When the terraform refresh command is run, Terraform reads the current state of the resources from the infrastructure provider, and then updates the state file with any changes that have occurred outside of Terraform.

upvoted 1 times

 **G4Exams** 1 year, 8 months ago

**Selected Answer: B**

refresh updates the state if there was a manual change in the infrastructure, for example via aws console.

upvoted 1 times

 **dani88ge** 1 year, 9 months ago

**Selected Answer: B**

It's B.

upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

<https://www.terraform.io/cli/commands/refresh>

upvoted 1 times

 **bora4motion** 1 year, 10 months ago

**Selected Answer: B**

I go with B

upvoted 3 times

Which of the following is not valid source path for specifying a module?

- A. source = "./module!version=v1.0.0"
- B. source = "github.com/hashicorp/example?ref=v1.0.0"
- C. source = "./module"
- D. source = "hashicorp/consul/aws"

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **depal\_dhir** Highly Voted 1 year, 10 months ago

**Selected Answer: A**

You don't use pipe "|" in the path  
upvoted 5 times

 **Bere** Most Recent 5 months, 4 weeks ago

**Selected Answer: A**

Valid usage for a local path:

```
module "example" {
 source = "./module"
 # Versioning for local paths is not applicable. You would control versioning with version control systems.
}
```

upvoted 1 times

 **Zeppoonstream** 1 year, 5 months ago

A. source = "./module!version=v1.0.0" is not a valid source path for specifying a module.

When specifying a module, a valid source path must include the module name and the provider, if specified. In this case, "./module!version=v1.0.0" is not a valid source path because it's missing the module name and "!" is an invalid character for a version.  
upvoted 3 times

 **G4Exams** 1 year, 8 months ago

**Selected Answer: A**

A tricky question but ./ means its a local path and there are no versions in local modules so must be A I guess.  
upvoted 4 times

 **Hizumi** 1 year, 10 months ago

Answer is A.  
All other choices are the correct way to input the path.  
Reference:  
upvoted 1 times

Question #153

*Topic 1*

Which of the following is true about terraform apply? (Choose two.)

- A. It only operates on infrastructure defined in the current working directory or workspace
- B. You must pass the output of a terraform plan command to it
- C. Depending on provider specification, Terraform may need to destroy and recreate your infrastructure resources
- D. By default, it does not refresh your state file to reflect current infrastructure configuration
- E. You cannot target specific resources for the operation

**Correct Answer:** AC

*Community vote distribution*

AC (100%)

 **camps** 1 year, 3 months ago

**Selected Answer: AC**

- A. It only operates on infrastructure defined in the current working directory or workspace.
- C. Depending on provider specification, Terraform may need to destroy and recreate your infrastructure resources.

The terraform apply command in Terraform is used to create or modify infrastructure resources according to the Terraform configuration in the current working directory or workspace.

upvoted 2 times

 **Zeppoonstream** 1 year, 5 months ago

- A. It only operates on infrastructure defined in the current working directory or workspace
- C. Depending on provider specification, Terraform may need to destroy and recreate your infrastructure resources

The following is true about terraform apply:

A. It only operates on infrastructure defined in the current working directory or workspace. Terraform apply command will only perform changes to the resources defined in the Terraform configuration files in the current working directory or workspace.

C. Depending on provider specification, Terraform may need to destroy and recreate your infrastructure resources: Terraform may need to destroy and recreate some of the resources if the provider requires it to apply changes, this is also known as "create-before-destroy" strategy

upvoted 2 times

 **G4Exams** 1 year, 8 months ago

**Selected Answer: AC**

A and C

upvoted 2 times

 **dani88ge** 1 year, 9 months ago

**Selected Answer: AC**

AC, for sure

upvoted 2 times

Question #154

Topic 1

Which of the following statements about local modules is incorrect?

- A. Local modules are not cached by terraform init command
- B. Local modules are sourced from a directory on disk
- C. Local modules support versions
- D. All of the above (all statements above are incorrect)
- E. None of the above (all statements above are correct)

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **rotimislaw** Highly Voted 1 year, 4 months ago

Why not A? Are local modules cached or not?

upvoted 5 times

 **LemonadeSoftware** Most Recent 8 months ago

The correct answer is C. "Local modules support versions" is incorrect.

A. This statement is incorrect. Local modules are cached by the terraform init command. When you run terraform init, Terraform downloads and installs the modules specified in the configuration, including local modules, and caches them in the .terraform directory.

B. This statement is correct. Local modules are sourced from a directory on disk. You can specify the path to the directory containing your local module in your Terraform configuration.

C. This statement is incorrect. Local modules do not support versions in the same way remote modules hosted on version control systems (e.g. Git) do. Local modules are typically referenced by a relative or absolute path, and there is no versioning mechanism built into Terraform for local modules.

D. This cannot be the correct answer because Statement B is correct.

E. This cannot be the correct answer because Statement C is incorrect.

So, the correct answer is C.

upvoted 3 times

 **camps** 1 year, 3 months ago

Selected Answer: C

C. Local modules support versions.

Local modules are sourced from a directory on disk and are typically used to organize and reuse Terraform code within a project. However, local modules do not support versions. This means that if you make changes to a local module, those changes will be reflected in all resources that use that module, regardless of when those resources were created.

upvoted 1 times

 **dinesh198728** 1 year, 10 months ago

Selected Answer: C

Version constraints are supported only for modules installed from a module registry, such as the public Terraform Registry or Terraform Cloud's private module registry. Other module sources can provide their own versioning mechanisms within the source string itself, or might not support versions at all. In particular, modules sourced from local file paths do not support version; since they're loaded from the same source repository, they always share the same version as their caller.

upvoted 3 times

 **depal\_dhir** 1 year, 10 months ago

Selected Answer: C

<https://www.terraform.io/language/modules/syntax>

upvoted 2 times

Question #155

Topic 1

Which of the following is true about Terraform's implementation of infrastructure as code? (Choose two.)

- A. It is only compatible with AWS infrastructure management
- B. You cannot reuse infrastructure configuration
- C. You can version your infrastructure configuration
- D. It requires manual configuration of infrastructure resources
- E. It allows you to automate infrastructure provisioning

**Correct Answer: BD**

*Community vote distribution*

CE (100%)

 **RVivek** Highly Voted 1 year, 9 months ago

**Selected Answer: CE**

- A- It supports hundreds of providers
- B- You can have version number on your files
- D-- manual provisioning defeats the very purpose of IaC

upvoted 9 times

 **camps** Most Recent 1 year, 3 months ago

**Selected Answer: CE**

- C. You can version your infrastructure configuration.

- E. It allows you to automate infrastructure provisioning.

Terraform's implementation of infrastructure as code has the following characteristics:

Option C is true - you can version your infrastructure configuration using version control systems like Git. Terraform supports multiple version control backends such as Git, Subversion, and Mercurial, and allows users to manage multiple versions of infrastructure code in the same repository.

Option E is true - Terraform allows you to automate infrastructure provisioning. Terraform uses configuration files to describe the desired state infrastructure, and then automatically provisions and configures the infrastructure to match that state.

upvoted 3 times

 **Manguu** 1 year, 3 months ago

**Selected Answer: CE**

No doubt about it

upvoted 1 times

 **Zepponstream** 1 year, 5 months ago

- C. You can version your infrastructure configuration
- E. It allows you to automate infrastructure provisioning

The following is true about Terraform's implementation of infrastructure as code:

C. You can version your infrastructure configuration: Terraform uses configuration files written in HashiCorp Configuration Language (HCL) to define infrastructure resources. These files can be versioned using a version control system (VCS) such as Git, allowing you to track changes to your infrastructure over time and roll back to previous versions if necessary.

E. It allows you to automate infrastructure provisioning: Terraform's infrastructure as code approach allows you to automate the provisioning of infrastructure resources. By defining the desired state of your infrastructure in code, you can use Terraform to create, modify, and delete resources in an automated and repeatable way.

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: CE**

C and E for sure

upvoted 1 times

 **smling** 1 year, 10 months ago

**Selected Answer: CE**

I go for CE.

upvoted 4 times

 **zyxphreez** 1 year, 10 months ago

**Selected Answer: CE**

C, E are correct

upvoted 3 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: CE**

It has to be CE.

upvoted 3 times

You need to write some Terraform code that adds 42 firewall rules to a security group as shown in the example.

```
resource "aws_security_group" "many_rules" {
 name = "many-rules"
 ingress {
 from_port = 443
 to_port = 443
 protocol = "tcp"
 cidr_blocks = "0.0.0.0/0"
 }
}
```

What can you use to avoid writing 42 different nested ingress config blocks by hand?

- A. A count loop
- B. A for block
- C. A for each block
- D. A dynamic block

**Correct Answer: D**

*Community vote distribution*

D (92%)

8%

 **Hizumi** Highly Voted 1 year, 10 months ago

Answer is D.

A dynamic block acts much like a for expression, but produces nested blocks instead of a complex typed value. It iterates over a given complex value, and generates a nested block for each element of that complex value.

Reference: <https://www.terraform.io/language/expressions/dynamic-blocks>

upvoted 7 times

 **Bere**  5 months, 4 weeks ago

**Selected Answer: D**

```
resource "aws_security_group" "many_rules" {
 name = "many_rules"

 dynamic "ingress" {
 for_each = var.ingress_rules
 content {
 from_port = ingress.value.from_port
 to_port = ingress.value.to_port
 protocol = ingress.value.protocol
 cidr_blocks = ingress.value.cidr_blocks
 }
 }
}

variable "ingress_rules" {
 description = "A list of ingress rules"
 type = list(object({
 from_port = number
 to_port = number
 protocol = string
 cidr_blocks = list(string)
 }))
 default = [
 {
 from_port = 80
 to_port = 80
 protocol = "tcp"
 cidr_blocks = ["0.0.0.0/0"]
 },
 # Add 41 more rules here...
]
}
```

upvoted 5 times

 **InformationOverload**  1 year, 6 months ago

**Selected Answer: D**

yep, D is correct.

upvoted 1 times

 **Rugaroo** 1 year, 6 months ago

**Selected Answer: D**

Some additional information comparing each of the options: <https://awstip.com/terraform-for-vs-for-each-7ff8506a1f94>

Question #157

Topic 1

Which of the following is the safest way to inject sensitive values into a Terraform Cloud workspace?

- A. Write the value to a file and specify the file with the -var-file flag
- B. Set a value for the variable in the UI and check the "Sensitive" check box
- C. Edit the state file directly just before running terraform apply
- D. Set the variable value on the command line with the -var flag

**Correct Answer: B**

*Community vote distribution*

B (88%)

13%

✉️  **kiran15789** 1 year, 2 months ago

**Selected Answer: A**

The safest way to inject sensitive values into a Terraform Cloud workspace is to write the value to a file and specify the file with the -var-file flag. Option A is the correct answer.

This method allows you to store the sensitive value in a file that can be encrypted and stored securely. Terraform Cloud supports encrypted variables, and you can encrypt the file that contains the sensitive value using a tool like SOPS or Vault before uploading it to Terraform Cloud.  
upvoted 1 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer: B**

B. Set a value for the variable in the UI and check the "Sensitive" check box.

When working with Terraform Cloud workspaces, the safest way to inject sensitive values into a Terraform Cloud workspace is to set a value for the variable in the UI and check the "Sensitive" check box. This will ensure that the value is stored securely and not visible in plain text in the Terraform Cloud UI or API.

Option A, writing the value to a file and specifying the file with the -var-file flag, may be less secure because the file could potentially be accessed by unauthorized users.

Option C, editing the state file directly just before running terraform apply, is not a best practice and could result in data loss or corruption.

Option D, setting the variable value on the command line with the -var flag, could result in the sensitive value being stored in plain text in the command history or other logs, which could be accessed by unauthorized users.

upvoted 4 times

✉️  **princajen** 1 year, 4 months ago

**Selected Answer: B**

B. Set a value for the variable in the UI and check the "Sensitive" check box is the safest way to inject sensitive values into a Terraform Cloud workspace. This ensures that the sensitive values are securely stored and encrypted in the workspace, and are not visible in the Terraform log state file. Writing the value to a file or setting the variable value on the command line can expose the sensitive data to unauthorized access, and editing the state file directly is not a recommended practice and can potentially corrupt the state.

upvoted 1 times

✉️  **wanrltw** 1 year, 7 months ago

<https://developer.hashicorp.com/terraform/cloud-docs/workspaces/variables/managing-variables#sensitive-values>

upvoted 4 times

✉️  **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

<https://www.terraform.io/cloud-docs/workspaces/variables>

upvoted 2 times

✉️  **Hizumi** 1 year, 10 months ago

Answer is B.

-var and -var-file overwrite workspace-specific and variable set variables that have the same key. From the workspace, variable can be added and checked off as being sensitive.

Reference: <https://www.terraform.io/cloud-docs/workspaces/variables/managing-variables#loading-variables-from-files>  
<https://www.terraform.io/cloud-docs/workspaces/variables>

upvoted 2 times

terraform apply will fail if you have not run terraform plan first to update the plan output.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Anil\_K786** 6 months ago

answer is B. False

upvoted 1 times

 **Pinky0289** 1 year, 10 months ago

Terraform plan is not mandatory. It just lays out the execution plan. It is an optional command. So, the answer is B. False

upvoted 3 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

Terraform plan is optional

upvoted 3 times

 **keiffo2** 1 year, 10 months ago

you don't need to do plan before you do apply

upvoted 2 times

How would you reference the attribute "name" of this fictitious resource in HCL?

```
resource "kubernetes_namespace" "example" {
 name = "test"
}
```

- A. resource.kubernetes\_namespace.example.name
- B. kubernetes\_namespace.test.name
- C. kubernetes\_namespace.example.name
- D. data.kubernetes\_namespace.name
- E. None of the above

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **Hizumi** Highly Voted 1 year, 10 months ago

Correct answer is C.

References to Resource Attributes Doc: <https://www.terraform.io/language/expressions/references#references-to-resource-attributes>  
upvoted 5 times

 **tycho** Most Recent 1 year, 2 months ago

Answer C; that would be the equivalent to kubectl create ns :)

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: C**

C looks fine to me!

upvoted 2 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: C**

<https://www.terraform.io/language/expressions/references#references-to-resource-attributes>

upvoted 3 times

A Terraform output that sets the "sensitive" argument to true will not store that value in the state file.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: B**

"Terraform will still record sensitive values in the state, and so anyone who can access the state data will have access to the sensitive values in cleartext."

upvoted 11 times

 **camps** Most Recent 1 year, 3 months ago

**Selected Answer: B**

B. False.

A Terraform output that sets the "sensitive" argument to true will still store that value in the state file. The "sensitive" argument is used to prevent the value from being displayed in plain text in the Terraform CLI output, Terraform Cloud UI, and other locations where output values may be displayed. However, the value will still be stored in the Terraform state file, which is used to track the current state of the infrastructure.

It is important to be aware that while the "sensitive" argument can help to prevent accidental exposure of sensitive values, it is not a substitute for proper security practices such as role-based access control and data encryption.

upvoted 3 times

 **BilalIg93350** 1 year, 4 months ago

This statement is true. When you set the sensitive argument to true in a Terraform output, the output value will not be shown in plain text in the Terraform state file or in Terraform command output. This is useful for sensitive information such as passwords, keys, or other secrets.

While the value of the output is still stored in the state file, it is stored in a hashed format that is not easily readable. Additionally, the value will not be shown in plain text when running the terraform output command or when viewing the state file.

Setting the sensitive argument to true in a Terraform output can help prevent sensitive information from being accidentally exposed or leaked.

upvoted 1 times

 **Loopjoke** 1 year, 5 months ago

Y'all wrong. Answer A and ChatGPT has as True as well.

upvoted 1 times

 **lezgino** 1 year, 5 months ago

The answer is B. ChatGPT also can say wrong answer. It should learn first)

upvoted 6 times

 **nakikoo** 1 year, 7 months ago

**Selected Answer: B**

it still stores in statefile, but is hidden for security when accessing provider information that requires sensitive information such as credentials

upvoted 1 times

 **G4Exams** 1 year, 8 months ago

**Selected Answer: B**

When we mark something as sensitive in the configuration then it's hidden from the CLI for example when an output is set. But it will still be present in the state file.

upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

<https://www.terraform.io/language/values/outputs>

upvoted 3 times

 **Hizumi** 1 year, 10 months ago

Answer is B.

Reference: <https://www.terraform.io/language/values/outputs>

upvoted 2 times

Question #161

Topic 1

Which are forbidden actions when the Terraform state file is locked? (Choose three.)

- A. terraform destroy
- B. terraform fmt

- C. terraform state list
- D. terraform apply
- E. terraform plan
- F. terraform validate

**Correct Answer: ADE**

*Community vote distribution*

ADE (100%)

 **camps** 1 year, 3 months ago

**Selected Answer: ADE**

- A. terraform destroy
- D. terraform apply

Question #162

Topic 1

Terraform installs its providers during which phase?

- A. Plan
- B. Init
- C. Refresh
- D. All of the above

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: B**

Trivial. B is correct  
upvoted 2 times

 **G4Exams** 1 year, 8 months ago

**Selected Answer: B**

Init for sure  
upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

Terraform installs providers at init phase  
upvoted 2 times

 **keiffo2** 1 year, 10 months ago

Providers are installed in the init phase - answer is correct  
upvoted 1 times

**Selected Answer: ADE**

Question #163

Topic 1

When does Sentinel enforce policy logic during a Terraform Enterprise run?

- A. Before the plan phase
- B. During the plan phase
- C. Before the apply phase
- D. After the apply phase

**Correct Answer: C**

*Community vote distribution*

C (88%)

12%

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

"Enforcing policy checks on runs - Policies are checked when a run is performed, after the terraform plan but before it can be confirmed or the terraform apply is executed."

upvoted 7 times

 **mohamed1999** Most Recent 6 months, 2 weeks ago

**Selected Answer: C**

Terraform Enterprise enforces Sentinel policies between the plan and apply phases of a run, preventing out of policy infrastructure from being provisioned. Unless overridden by an authorized user, only plans that pass all Sentinel policies checked against them are allowed to proceed to the apply step.

upvoted 3 times

 **zanhsieh** 1 year, 2 months ago

**Selected Answer: C**

C. See the official diagram here:  
<https://developer.hashicorp.com/terraform/tutorials/policy/sentinel-install>

upvoted 2 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

B. During the plan phase.

Sentinel is a policy-as-code framework integrated with Terraform Enterprise that allows organizations to define and enforce policies on infrastructure changes. Sentinel enforces policy logic during the plan phase of a Terraform Enterprise run, before any changes are applied to the infrastructure. During the plan phase, Terraform generates an execution plan that describes the changes that will be made to the infrastructure. Sentinel evaluates policy rules against this execution plan to determine whether the proposed changes comply with the defined policies. If any violations are detected, the plan is rejected, and the changes are not applied.

Sentinel does not enforce policy logic before the plan phase or after the apply phase. However, Sentinel policies can also be used to enforce compliance on policy requirements that are not directly related to infrastructure changes, such as resource tagging or naming conventions. In these cases, Sentinel policies may be evaluated at other points in the Terraform Enterprise workflow, such as during VCS (version control system) integration or during cost estimation.

upvoted 2 times

 **mohamed1999** 6 months, 2 weeks ago

Terraform Enterprise enforces Sentinel policies between the plan and apply phases of a run, preventing out of policy infrastructure from being provisioned. Unless overridden by an authorized user, only plans that pass all Sentinel policies checked against them are allowed to proceed to the apply step.

upvoted 1 times

 **phidelics** 1 year, 3 months ago

**Selected Answer: C**

terraform plan >>>>sentinel policy>>>>>>terraform apply

upvoted 3 times

 **Pinky0289** 1 year, 10 months ago

Policies are enforced after the plan and before the apply commands. so, the answer is option C.

upvoted 1 times

What is the purpose of a Terraform workspace in either open source or enterprise?

- A. Workspaces allow you to manage collections of infrastructure in state files
- B. A logical separation of business units
- C. A method of grouping multiple infrastructure security policies
- D. Provides limited access to a cloud environment

**Correct Answer: A**

*Community vote distribution*

A (90%)

10%

 **camps** 1 year, 3 months ago

**Selected Answer: A**

- A. Workspaces allow you to manage collections of infrastructure in state files.

The purpose of a Terraform workspace, both in open source and enterprise, is to allow users to manage multiple "instances" of infrastructure within the same configuration codebase. Each workspace is a separate instance of a Terraform state file that can be managed independently. This means that a single Terraform configuration can be used to manage multiple sets of resources, with each set of resources represented by separate workspace.

Workspaces are useful when there is a need to manage multiple versions of the same infrastructure in the same configuration codebase, such different environments (dev, staging, prod) or different regions. By creating a separate workspace for each instance of infrastructure, users can manage them independently without causing conflicts or overwriting state.

Option B, C, and D are not correct. While workspaces may be used in conjunction with logical separation of business units, grouping multiple infrastructure security policies, or providing limited access to a cloud environment, they are not the primary purpose of workspaces in Terraform.

upvoted 2 times

 **princajen** 1 year, 4 months ago

**Selected Answer: A**

A. Workspaces allow you to manage collections of infrastructure in state files. The purpose of Terraform workspaces is to manage multiple state files for the same Terraform configuration. Workspaces provide a way to organize and isolate state files for different environments (e.g., development, staging, production) or different configurations (e.g., different regions, different sets of resources). Each workspace has its own copy of the state file, allowing changes to be made independently and preventing conflicts. Workspaces are available in both open source and Terraform Enterprise.

Option B, C, and D are not accurate descriptions of the purpose of Terraform workspaces. Workspaces do not provide logical separation of business units, grouping of infrastructure security policies, or limited access to a cloud environment. These are functions of other Terraform Enterprise features, such as organizations, policy sets, and workspaces with role-based access controls.

upvoted 3 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

A workspace is a logical separation of resources within the same configuration. Workspaces allow you to manage multiple environments, such as production, staging, and development, within the same configuration and state file. I will go for A here.

upvoted 1 times

 **macross** 1 year, 7 months ago

Workspaces in Terraform are simply independently managed state files. A workspace contains everything that Terraform needs to manage a given collection of infrastructure, and separate Workspaces function like completely separate working directories. We can manage multiple environments with Workspaces. Answer A

upvoted 1 times

✉  **secdaddy** 1 year, 7 months ago

There is no discussion of business units in the documentation so (A) seems to make the most sense, based on these two sources both of which mention state :

Terraform CLI workspaces are associated with a specific working directory and isolate multiple state files in the same working directory, letting you manage multiple groups of resources with a single configuration.  
<https://developer.hashicorp.com/terraform/cloud-docs/workspaces>

The persistent data stored in the backend belongs to a workspace. The backend initially has only one workspace containing one Terraform state file associated with that configuration. Some backends support multiple named workspaces, allowing multiple states to be associated with a single configuration. The configuration still has only one backend, but you can deploy multiple distinct instances of that configuration without configuring a new backend or changing authentication credentials.

<https://developer.hashicorp.com/terraform/language/state/workspaces>

upvoted 2 times

✉  **BleHi** 1 year, 7 months ago

**Selected Answer: B**

<https://www.hashicorp.com/resources/why-consider-terraform-enterprise-over-open-source>

Terraform Enterprise provides a logical unit to break down infrastructure as code into workspaces.

upvoted 1 times

✉  **yogishrb2020** 1 year, 9 months ago

**Selected Answer: A**

<https://www.terraform.io/cloud-docs/workspaces>

upvoted 3 times

✉  **yaza** 1 year, 9 months ago

Selected Answer: B

upvoted 1 times

✉  **yaza** 1 year, 9 months ago

because A is the correct of cli-workspace and not cloud workspace

upvoted 1 times

✉  **Hizumi** 1 year, 10 months ago

Answer is A

Working with Terraform involves managing collections of infrastructure resources, and most organizations manage many different collections. Reference: <https://www.terraform.io/cloud-docs/workspaces> (Refer to workspace contents)

upvoted 2 times

✉  **SanderIsTheBestCloudShaper** 1 year, 10 months ago

hello , is this dump valid ? i see u really active can u answer me.

upvoted 2 times

✉  **kashifhussain** 1 year, 10 months ago

I would like to know that as well please.

upvoted 1 times

Question #165

Topic 1

Which is the best way to specify a tag of v1.0.0 when referencing a module stored in Git (for example git::<https://example.com/vpc.git>)?

- A. Append ?ref=v1.0.0 argument to the source path
- B. Add version = "1.0.0" parameter to module block
- C. Nothing - modules stored on GitHub always default to version 1.0.0

D. Modules stored on GitHub do not support versioning

**Correct Answer: A**

*Community vote distribution*

A (84%)

B (16%)

✉  **RVivek** Highly Voted 1 year, 9 months ago

**Selected Answer: A**

If the source is from terraform registry then we can use version = "1.0.0"  
when using git as the source we have to use ?ref=1.0.0 the exact git path given in the question is given in this terraform official documentation  
please check <https://www.terraform.io/language/modules/sources#selecting-a-revision>

upvoted 5 times

✉  **BaburTurk** Most Recent 1 year, 6 months ago

**Selected Answer: A**

example  
source = "git::https://example.com/vpc.git?ref=v1.2.0"

upvoted 3 times

✉  **dinesh198728** 1 year, 10 months ago

**Selected Answer: A**

A is correct

upvoted 3 times

✉  **zyxphreez** 1 year, 10 months ago

**Selected Answer: A**

as Donathon said....  
<https://www.terraform.io/language/modules/sources#selecting-a-revision>

please check the module repository is GIT.... Burakko answer is incorrect

upvoted 2 times

✉  **Burakko** 1 year, 10 months ago

There was exactly the same question earlier. This was the answer. however if you ask me, both answers must be correct. Because they both work.

upvoted 1 times

✉  **donathon** 1 year, 10 months ago

**Selected Answer: A**

<https://www.terraform.io/language/modules/sources#selecting-a-revision>

upvoted 3 times

✉  **Burakko** 1 year, 10 months ago

**Selected Answer: B**

Use the version argument in the module block to specify versions:

Example:

```
module "consul" {
 source = "hashicorp/consul/aws"
 version = "0.0.5"
```

```
servers = 3
}
```

upvoted 3 times

✉  **Linus11** 1 year, 9 months ago

This is about version stored in GIT, not in consul.

upvoted 2 times

Changing the Terraform backend from the default "local" backend to a different one after doing your first terraform apply is:

- A. Mandatory
- B. Optional
- C. Impossible
- D. Discouraged

**Correct Answer: B**

*Community vote distribution*

B (100%)

✉️  **szyszka** 1 year, 4 months ago

I think most of you misread the question. It is not about requirement of running terraform apply it is about whether it is possible to change backend after first apply command.

upvoted 1 times

✉️  **Paul\_white** 11 months, 2 weeks ago

SO WHAT IS THE ANSWER ?

upvoted 1 times

✉️  **adouban** 1 year, 7 months ago

**Selected Answer: B**

B. Optional , the mandatory using terraform init

upvoted 1 times

✉️  **G4Exams** 1 year, 8 months ago

**Selected Answer: B**

You can do it. Not a must. If it's useful depends on the usecase. So B.

upvoted 2 times

✉️  **itsVespucci** 1 year, 8 months ago

its gotta be C as there will be an error.

When you change a backend's configuration, you must run terraform init again to validate and configure the backend before you can perform plans, applies, or state operations.

upvoted 1 times

✉️  **wanrltw** 1 year, 7 months ago

This has nothing to do with the question. Can you change the backend after running your first `terraform apply`? Yes, you can. Sure you'll have to run the init again but that's another story - question doesn't mention anything about "after".

upvoted 2 times

✉️  **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

It is optional

upvoted 2 times

✉️  **keiffo2** 1 year, 10 months ago

It is Optional - so it's B

upvoted 1 times

Question #167

*Topic 1*

You have modified your local Terraform configuration and ran `terraform plan` to review the changes. Simultaneously, your teammate manually modified the infrastructure component you are working on. Since you already ran `terraform plan` locally, the execution plan for `terraform apply` will be the same.

- A. True
- B. False

**Correct Answer:** *B*

*Community vote distribution*

B (100%)

 **dепал\_dхир**  1 year, 10 months ago

**Selected Answer: B**

Your plan has been changed. So its best to run the plan again to make sure the desired state is achieved.  
upvoted 6 times

 **Bere**  5 months, 3 weeks ago

**Selected Answer: B**

The execution plan created by terraform plan is based on the state at the time terraform plan was run.  
If the actual infrastructure has been manually modified by your teammate after you ran terraform plan, then the infrastructure will have diverge from what Terraform believes it to be.  
When you subsequently run terraform apply, Terraform will create a new execution plan based on the current state of the infrastructure, not or the plan you previously viewed.  
This means the execution plan during apply could be different if the infrastructure has been changed outside of Terraform after the initial plan command was executed.  
It's always a good practice to run terraform plan again just before terraform apply, especially in environments where manual changes to the infrastructure might occur.

upvoted 4 times

 **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: B**

To ensure that the execution plan you reviewed with terraform plan is the exact one used by terraform apply, you can save the plan to a file and pass that file to terraform apply. For example:  
terraform plan -out=my\_plan.tfplan  
terraform apply my\_plan.tfplan

upvoted 3 times

 **chael188** 1 year, 5 months ago

**Selected Answer: B**

When you run Apply, it will check if the infrastructure has changed. In this case, the other person made that change.  
upvoted 1 times

 **senthil5000** 1 year, 6 months ago

**Selected Answer: B**

Running terraform plan is optional. Even if we run terraform apply directly without running terraform plan before, it will check for infrastructure before applying.  
upvoted 1 times

 **Rugaroo** 1 year, 7 months ago

Question #168

Topic 1

efc  
de

terraform apply is failing with the following error. What next step should you take to determine the root cause of the problem?

Error loading state: AccessDenied: Access Denied status code: 403, request id: 288766CE5CCA24A0, host id: FOOBAR

- A. Set TF\_LOG=DEBUG
- B. Review syslog for Terraform error messages
- C. Run terraform login to reauthenticate with the provider
- D. Review /var/log/terraform.log for error messages

**Correct Answer: A**

*Community vote distribution*

A (80%)

C (20%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: A**

Terraform has detailed logs which can be enabled by setting the TF\_LOG environment variable to any value. This will cause detailed logs to appear on stderr.

You can set TF\_LOG to one of the log levels (in order of decreasing verbosity) TRACE, DEBUG, INFO, WARN or ERROR to change the verbosity of the logs.

upvoted 6 times

 **Spandrop** Most Recent 7 months ago

"A" maybe make sense, but why you would like to enable TF\_LOG for a message that is crystal clear: ACCESS DENIED?!

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

A makes sense here.

upvoted 1 times

 **DerekKey** 1 year, 7 months ago

A Correct

C - no - provider is not responsible for managing state

upvoted 1 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: A**

to determine the root cause of the problem , we need to first enable logging

upvoted 1 times

 **jednorozec2022** 1 year, 9 months ago

A

A. Set TF\_LOG=DEBUG >> only this one sound like correct answer

B. Review syslog for Terraform error messages >> firstly you need to turn on terraform logging  
ref <https://www.terraform.io/internals/debugging>

C. Run terraform login to reauthenticate with the provider >> login is used to connect with Terraform Cloud, Terraform Enterprise, or any other host that offers Terraform services  
ref <https://www.terraform.io/cli/commands/login>

D. Review /var/log/terraform.log for error messages >> you would have to turn on logging and add path to logging  
ref <https://www.terraform.io/internals/debugging>

upvoted 4 times

 **jjkcoins** 1 year, 10 months ago

**Selected Answer: C**

Error loading state: AccessDenied: Access Denied status code: 403, request id: 288766CE5CCA24A0, host id: FOOBAR

Question #169

Topic 1

As a member of an operations team that uses infrastructure as code (IaC) practices, you are tasked with making a change to an infrastructure stack running in a public cloud.

Which pattern would follow IaC best practices for making a change?

- A. Clone the repository containing your infrastructure code and then run the code
- B. Use the public cloud console to make the change after a database record has been approved
- C. Make the change programmatically via the public cloud CLI
- D. Make the change via the public cloud API endpoint
- E. Submit a pull request and wait for an approved merge of the proposed changes

**Correct Answer: E**

*Community vote distribution*

E (100%)

 **camps** 1 year, 3 months ago

**Selected Answer: E**

E. Submit a pull request and wait for an approved merge of the proposed changes.

As a member of an operations team that uses infrastructure as code (IaC) practices, the best practice for making a change to an infrastructure stack running in a public cloud is to submit a pull request (PR) and wait for an approved merge of the proposed changes. This approach follows the best practice of using version control for infrastructure code and collaborating through pull requests to review and approve changes. By following this process, changes to the infrastructure stack are tracked, reviewed, and approved in a controlled and auditable way.

upvoted 1 times

 **fedeX** 1 year, 6 months ago

**Selected Answer: E**

Following IaC best practices, the recommended pattern for making a change to an infrastructure stack running in a public cloud would be to submit a pull request and wait for an approved merge of the proposed changes.

This approach allows you to use version control and review processes to ensure that changes to the infrastructure are carefully reviewed and tested before being applied. It also allows you to track changes over time and roll back any problematic changes if necessary.

Option A (cloning the repository and running the code) is a good first step, but it does not address the need for review and approval of the changes. Option B (using the public cloud console) is generally not recommended, as it bypasses the version control and review processes that are central to IaC best practices. Options C (making the change programmatically via the public cloud CLI) and D (making the change via the public cloud API endpoint) are both more in line with IaC best practices, as they allow you to automate the process of making changes, but they still do not address the need for review and approval of the changes.

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: E**

We should follow CI/CD best practices, the answer is E.

upvoted 2 times

 **dani88ge** 1 year, 9 months ago

**Selected Answer: E**

It's E

upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: E**

The best practice is PR (Pull Request)

upvoted 2 times

 **keiffo2** 1 year, 10 months ago

CI-CD best practices recommend pull requests i.e. merge the branch to main, in order to run a pipeline request - which kicks off a terraform deploy

upvoted 1 times

 **keiffo2** 1 year, 10 months ago

So Answer E

upvoted 1 times

What command can you run to generate DOT (Document Template) formatted data to visualize Terraform dependencies?

- A. terraform refresh
- B. terraform show
- C. terraform graph
- D. terraform output

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

The terraform graph command is used to generate a visual representation of either a configuration or execution plan. The output is in the DOT format, which can be used by GraphViz to generate charts.

upvoted 8 times

 **SilentMilli** Most Recent 1 year, 3 months ago

**Selected Answer: C**

The correct answer is C. terraform graph command generates DOT (Document Template) formatted data to visualize Terraform dependencies. The terraform graph command generates a visualization of Terraform resources as a directed acyclic graph (DAG) using the Graphviz DOT format. This visualization can be used to understand and debug complex resource dependencies in your infrastructure as code.

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: C**

C is correct, good explanation from @Burakko!

upvoted 1 times

Which provider authentication method prevents credentials from being stored in the state file?

- A. Using environment variables
- B. Specifying the login credentials in the provider block
- C. Setting credentials as Terraform variables
- D. None of the above

**Correct Answer: A**

*Community vote distribution*

A (55%)

D (45%)

 **Jayanth** Highly Voted 11 months, 3 weeks ago

D is the right answer.

I tested in my local machine to create Sql server with 2 environment variables \$env:TF\_VAR\_sql\_admin = "username" and \$env:TF\_VAR\_sql\_password = "sqldbpassword"

Also created the SQL Server with Terraform which accesses env variable during execution.

BUT FOUND MY SENSITIVE ENVIRONMENT VARIABLE VALUES ARE STILL LISTED IN THE "STATE FILE"

so answer should be "D"

upvoted 5 times

✉ **dzhang344** Most Recent 3 weeks, 1 day ago

**Selected Answer: A**

When you use environment variables to store credentials, Terraform does not include these credentials in the state file. Environment variables read at runtime, which means they are not persisted in the configuration files or the state file.

upvoted 1 times

✉ **abobeida94** 2 months ago

**Selected Answer: A**

Answer is A 100%: Using environment variables

We already use Terraform this way:

Bash

```
export AWS_ACCESS_KEY_ID="YOUR_ACCESS_KEY"
export AWS_SECRET_ACCESS_KEY="YOUR_SECRET_KEY"
```

Terraform

```
provider "aws" {
 access_key = "YOUR_ACCESS_KEY"
 secret_key = "YOUR_SECRET_KEY"
}
```

upvoted 1 times

✉ **mattuyghur** 4 months, 4 weeks ago

**Selected Answer: A**

Using environment variables

upvoted 1 times

✉ **Bere** 5 months, 3 weeks ago

**Selected Answer: D**

1. Code example:

```
...
resource "azurerm_sql_server" "example" {
 name = "example-sqlserver"
 resource_group_name = azurerm_resource_group.example.name
 location = azurerm_resource_group.example.location
 version = "12.0"
 administrator_login = var.sql_admin_username
 administrator_login_password = var.sql_admin_password
}
```

```
variable "sql_admin_username" {}
variable "sql_admin_password" {}
...
```

2. Set env variables:

```
export TF_VAR_sql_admin_username="adminuser"
export TF_VAR_sql_admin_password="SuperSecretPassword"
```

3. terraform init

4. terraform apply

5. After applying, if you inspect the state file (terraform.tfstate), you will find that it contains the administrator login and password.

upvoted 1 times

 **frees** 7 months ago

**Selected Answer: D**

Terraform Enterprise and Terraform Cloud credentials are not stored in Terraform state or the CI/CD platform. Therefore, the correct answer to your question is D. None of the above.

upvoted 1 times

 **AndreiWebNet** 5 months, 3 weeks ago

It doesn't say anything about Terraform Enterprise or Terraform Cloud. Its something you assumed. . .

upvoted 3 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: A**

nswer: using environment variables

The only method list above that will not result in the username/password being written to the state file is environment variables. All of the other options will result in the provider's credentials in the state file.

Terraform runs will receive the full text of sensitive variables, and might print the value in logs and state files if the configuration pipes the value through to an output or a resource parameter. Additionally, Sentinel mocks downloaded from runs will contain the sensitive values of Terraform (but not environment) variables. Take care when writing your configurations to avoid unnecessary credential disclosure. Whenever possible, use environment variables since these cannot end up in state files or in Sentinel mocks. (Environment variables can end up in log files if TF\_LOG is set to TRACE.)

upvoted 1 times

 **akm\_1010** 11 months, 1 week ago

**Selected Answer: D**

All secrets will end up in statefile.

upvoted 1 times

 **Alandt** 5 months, 3 weeks ago

Your answer is right, but your explanation is not.

upvoted 1 times

 **Alandt** 5 months, 3 weeks ago

Wrong.

In terraform, are environment variables stored in state file?

ChatGPT

No, environment variables are not stored in the Terraform state file. The state file contains information about resources, not configuration values. Use environment variables or other secure methods to pass sensitive information during Terraform execution.

upvoted 2 times

 **Rajmane** 11 months, 1 week ago

**Selected Answer: D**

It's D only currently there is no way to prevent it

upvoted 2 times

 **Ha\_BaruH\_Architect13** 11 months, 4 weeks ago

nothign prevents this, only thing is we can encrytp answer is D

upvoted 3 times

 **VSMu** 12 months ago

**Selected Answer: D**

Refer: <https://developer.hashicorp.com/terraform/language/values/variables>

Setting a variable as sensitive prevents Terraform from showing its value in the plan or apply output, when you use that variable elsewhere in your configuration.

Terraform will still record sensitive values in the state, and so anyone who can access the state data will have access to the sensitive values in cleartext. For more information, see Sensitive Data in State.

upvoted 2 times

 [Removed] 1 year ago

**Selected Answer: A**

The answer is A. Using environment variables.

Here is an example of how to use environment variables to provide authentication credentials for an AWS provider:  
provider "aws" {

```
region = var.aws_region
access_key_id = var.aws_access_key_id
secret_access_key = var.aws_secret_access_key
}
```

upvoted 1 times

 Jhaggar 1 year, 2 months ago

**Selected Answer: D**

No, environment variables are not safe to store credentials in the state file of Terraform. Environment variables can be accessed by any process running on the same machine, including potentially malicious processes. It's important to use a secure method of storing credentials, such as using a secrets manager or key vault. Additionally, it's important to ensure that the state file itself is properly secured, either by encrypting it or storing it in a secure location.

upvoted 2 times

 OMERKENT 1 year, 2 months ago

I think correct answer is A.

I have checked in my remote state file sitting in Azure storage account. (I used Azure DevOps environment variables) secret files are not visible in the state file.

upvoted 2 times

 zanhsieh 1 year, 2 months ago

**Selected Answer: A**

Opt A. If you look into official terraform provider documentation, including terraform enterprise, all providers point to "Dynamic Provider Credentials". This workflow generally exposes a temporary OIDC compliment token as environment variable and authenticated by cloud providers. So I would say the straight forward answer would be environment variables.

upvoted 3 times

 zanhsieh 1 year, 2 months ago

<https://developer.hashicorp.com/terraform/enterprise/workspaces/dynamic-provider-credentials>

upvoted 1 times

 FarziWaliMarzi 1 year, 3 months ago

**Selected Answer: D**

Definitely "D", I wonder how A is even an "authentication method"? Read the question carefully, or am I trying to read in between the lines  
upvoted 1 times

 Stanislav4907 1 year, 3 months ago

**Selected Answer: D**

None of the above options prevents credentials from being stored in the state file.

Storing credentials in Terraform code or environment variables is not recommended, as it can expose sensitive information and make it more difficult to manage and rotate credentials.

Instead, you should use an external authentication method, such as the "external" authentication method in Terraform, which allows you to execute an external program to obtain authentication credentials at runtime, rather than storing the credentials in the state file. This method keeps your credentials secure and allows you to use authentication mechanisms that do not expose credentials in plain text or that require interactive authentication.

upvoted 2 times

Running terraform fmt without any flags in a directory with Terraform configuration files will check the formatting of those files without changing their contents.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (67%)

A (33%)

 **Burakko**  1 year, 10 months ago

**Selected Answer: B**

The terraform fmt command is used to rewrite Terraform configuration files to a canonical format and style.  
upvoted 15 times

 **ARBAR**  2 months, 3 weeks ago

**Selected Answer: A**

Running terraform fmt without any flags in a directory with Terraform configuration files will indeed check the formatting of those files without changing their contents. It will only modify the formatting to comply with Terraform's style conventions if necessary, without altering the actual content of the files.

upvoted 1 times

 **Bere** 5 months, 3 weeks ago

**Selected Answer: B**

Running terraform fmt without any flags in a directory with Terraform configuration files will not only check the formatting of those files but will also rewrite them to a canonical format if they are not already formatted correctly.

If you want to check the formatting without making any changes, you should use the -check flag with the command. This flag will instruct Terraform to check if the input is formatted correctly and return a non-zero exit code if it isn't, without modifying the files.

upvoted 3 times

 **Albion** 7 months, 1 week ago

**Selected Answer: B**

Tested. The terraform fmt command, format files content. No need for any flag  
upvoted 1 times

 **diegoa** 7 months, 2 weeks ago

**Selected Answer: A**

If no flag is given, fmt rewrites the Terraform configuration files to a canonical format and style.<https://developer.hashicorp.com/terraform/cli/commands/fmt>  
upvoted 2 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: A**

just read the document, it says if you use it without a flag or directory it will not process anything. however if you add a flag it will process it  
upvoted 1 times

 **LemonadeSoftware** 8 months ago

A 100%

Running terraform fmt without any flags in a directory with Terraform configuration files checks the formatting without altering the content because the command is used to format and standardize the layout of Terraform configuration files according to the defined conventions. When executed without any flags, it checks whether the files comply with the expected formatting standards but does not modify the content unless there's a need for formatting changes to align with the defined conventions.

If you want to format it, you have to add the "-w" flag  
upvoted 2 times

 **arun00028** 9 months ago

Option A  
upvoted 1 times

 **VSMu** 12 months ago

**Selected Answer: B**

Tested. The answer is B. Running terraform fmt updates (formats) the .tf files in the current working directory.  
upvoted 1 times

 **SimoAz** 12 months ago

**Selected Answer: A**

default write value is false donc A is correct  
upvoted 2 times

 **tycho** 1 year, 2 months ago

apparently according to this question the changing of the format is a change in the file content, therefore answer is B ; somehow it wants to highlight the diff between -check option and without it  
upvoted 1 times

 **BennaniHaythem** 1 year, 3 months ago

True. Running terraform fmt without any flags will check the formatting of the Terraform configuration files in the current directory, and it will update the files to conform to the standard formatting rules defined by Terraform. However, it will not change the contents of the files. If you want to modify the contents of the files to match the formatting rules, you can run terraform fmt --write=true.

upvoted 3 times

 **Stanislav4907** 1 year, 3 months ago

**Selected Answer: A**

Yes, that's correct. Running terraform fmt without any flags in a directory with Terraform configuration files will check the formatting of those files without changing their contents.

terraform fmt is a command that formats your Terraform configuration files according to a set of standard conventions. This makes your code more readable, easier to maintain, and more consistent with best practices.

upvoted 2 times

 **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: B**

If you want to check the formatting without modifying the files, you can use the -check flag:  
terraform fmt -check

upvoted 2 times

 **princajen** 1 year, 4 months ago

**Selected Answer: A**

A. True.

Running terraform fmt without any flags in a directory with Terraform configuration files will check the formatting of those files without changing their contents. The command will print a list of files that need formatting changes, but it will not actually make any changes to the files. This can be useful for checking if files have been manually edited or if they need to be updated to conform to a new format.

If you want to apply the formatting changes to the files, you can use the -write flag or the -check flag to check and write the formatting changes at the same time.

upvoted 1 times

 **David\_C\_90** 1 year, 4 months ago

This is wrong.

\$ terraform fmt -h  
Rewrites all Terraform configuration files to a canonical format. (...)

---  
If you want to check without changing the files, use -write=false or -check

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: B**

Tested in lab, B is correct.

upvoted 3 times

 **sameed** 1 year, 7 months ago

**Selected Answer: B**

It will not only check the content but reformat it if required.

upvoted 1 times

Question #173

Topic 1

terraform init retrieves the source code for all referenced modules.

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

A (74%)

B (26%)

 **DevoteamAnalytix**  1 year, 1 month ago

**Selected Answer: A**

<https://developer.hashicorp.com/terraform/cli/commands/init#child-module-installation>

"During init, the configuration is searched for module blocks, and the source code for referenced modules is retrieved from the locations given their source arguments."

upvoted 6 times

 **ARBAR**  2 months, 3 weeks ago

**Selected Answer: B**

terraform init initializes a Terraform working directory by downloading any necessary plugins and modules specified in the configuration files. It doesn't retrieve the source code for all referenced modules directly. Instead, it initializes the modules based on the module configurations and retrieves their source code when Terraform needs to apply changes or perform other operations.

upvoted 1 times

 **Ajit18** 4 months, 1 week ago

B. False

Terraform init does not automatically retrieve the source code for all referenced modules.

Here's a breakdown of what terraform init actually does:

Initializes the Terraform working directory.

Downloads and installs required providers.

Sets up the necessary infrastructure for working with Terraform.

To retrieve module source code, you'd use `terraform get` or a separate module download mechanism like Git or HTTP. The `source` argument within your Terraform configuration specifies the location of the module's source code, which `terraform init` then uses to download the code during the installation process.

upvoted 2 times

 **Ganesh** 5 months, 1 week ago

B. False

terraform init does a lot of setup work, including initializing the backend, installing providers that are used in the configuration, and preparing a working directory for other commands. However, its role in handling modules is specific to initializing modules by downloading their source code if they are not already present locally and are referenced from remote sources. It doesn't retrieve the source code for modules that are already initialized or if the modules are defined locally (in the same repository and not requiring a download). So, while it does prepare modules for use, saying it retrieves the source code for "all" referenced modules might be misleading if interpreted to mean it re-downloads or updates module already initialized without the -upgrade flag.

upvoted 2 times

 **piezas** 1 year, 1 month ago

**Selected Answer: B**

B. False

The statement is false. The terraform init command does not automatically retrieve the source code for all referenced modules in Terraform configuration. Instead, terraform init is used to initialize a Terraform working directory by downloading and installing the required providers and setting up the necessary infrastructure to work with Terraform.

The source code of the modules should be present locally in the working directory or in a remote repository specified in the Terraform configuration. To retrieve the modules, you need to use the terraform get command or set up a module download mechanism like Git or HTTP.

upvoted 2 times

 **Zeppoonstream** 1 year, 5 months ago

**Selected Answer: A**

A. True. Terraform init retrieves the source code for all referenced modules. When Terraform init is run, it will automatically download the source code for any modules referenced in the configuration files. This ensures that the correct version of the module is available for use when Terraform applies changes to the infrastructure. The source code is then cached locally so that it can be used in future runs without having to re-download it. It's important to notice that this command also initializes the backend and sets the backend configuration if specified.

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

Yes, the terraform init command retrieves the source code for all referenced modules in a Terraform configuration. It does not download the source code for any unreferenced modules or plugins.

upvoted 1 times

 **yogishrb2020** 1 year, 9 months ago

**Selected Answer: A**

<https://www.terraform.io/cli/commands/init>

upvoted 2 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: B**

If the referenced module is a local module then Terraform init does not retrieve the module or its source code.

upvoted 1 times

 **itsVespucci** 1 year, 8 months ago

it will still retrieve it but would not cache it

tf docs: During init, the configuration is searched for module blocks, and the source code for referenced modules is retrieved from the locations given in their source arguments.

upvoted 1 times

 **omodara** 1 year, 9 months ago

<https://www.terraform.io/language/modules/sources>

Terraform uses this during the module installation step of terraform init to download the source code to a directory on local disk so that other Terraform commands can use it. It is A

upvoted 4 times

 **lordogre16** 1 year, 9 months ago

**Selected Answer: A**

During init, the configuration is searched for module blocks, and the source code for referenced modules is retrieved from the locations given in their source arguments.

<https://www.terraform.io/cli/commands/init>

upvoted 2 times

 **A\_A\_AB** 1 year, 9 months ago

**Selected Answer: B**

That's False. Init is just for installing the providers source code, not the modules.

upvoted 1 times

 **RVivek** 1 year, 9 months ago

<https://www.terraform.io/language/modules/sources>

Module source codes also

upvoted 2 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: A**

Terraform installs providers, initialises source code & modules etc at this stage

upvoted 1 times

You have a Terraform configuration that defines a single virtual machine with no references to it. You have run `terraform apply` to create the resource, and then removed the resource definition from your Terraform configuration file.

What will happen when you run `terraform apply` in the working directory again?

- A. Nothing
- B. Terraform will destroy the virtual machine
- C. Terraform will error
- D. Terraform will remove the virtual machine from the state file, but the resource will still exist

**Correct Answer: B***Community vote distribution*

B (100%)

**✉️**  **Nick\_001** 1 year, 4 months ago**Selected Answer: B**

tested in lab. When reference to resource is removed in config file the next "terraform apply" will destroy the resource. Below is the output from cli  
# aws\_instance.name will be destroyed  
# (because aws\_instance.name is not in configuration)  
upvoted 2 times

**✉️**  **dokaedu** 1 year, 9 months ago

The question is saying removing resource definition of the resource, NOT removing the resource (VM)  
upvoted 1 times

**✉️**  **dokaedu** 1 year, 9 months ago

The question is saying removing the description of the resource, NOT removing the resource (VM)  
upvoted 1 times

**✉️**  **dnscloud02** 1 year, 9 months ago

definition not description  
upvoted 1 times

**✉️**  **depal\_dhir** 1 year, 10 months ago**Selected Answer: B**

This will destroy the VM  
upvoted 4 times

**✉️**  **keiffo2** 1 year, 10 months ago

If you remove the resource from your config file and the resource is in your state file, terraform will apply the configuration in the config file - which is to delete the resource  
upvoted 4 times

Which configuration consistency errors does `terraform validate` report?

- A. A mix of spaces and tabs in configuration files
- B. Differences between local and remote state
- C. Terraform module isn't the latest version

D. Declaring a resource identifier more than once

**Correct Answer: D**

*Community vote distribution*

D (88%)

13%

✉️  **Bere** 5 months, 3 weeks ago

**Selected Answer: D**

1. Code example:

```
resource "aws_instance" "example" {
ami = "ami-123456"
instance_type = "t2.micro"
}
```

```
resource "aws_instance" "example" {
ami = "ami-654321"
instance_type = "t2.micro"
}
```

2. terraform validate

3. Terraform will report an error indicating that a resource with the same name is declared more than once.

upvoted 2 times

✉️  **kiran15789** 1 year, 2 months ago

**Selected Answer: A**

The terraform validate command checks the syntax and validates the configuration files in a Terraform module. It does not check for configuration consistency errors like differences between local and remote state or outdated module versions.

Question #176

Topic 1

In Terraform HCL, an object type of object({ name=string, age=number }) would match this value:

A.

```
{
 name = "John"
 age = fifty two
}
```

B.

```
{
 name = "John"
 age = 52
}
```

**Correct Answer: B**

✉️  **cankayahmet** Highly Voted 1 year ago

what a stupid question!

upvoted 7 times

✉️  **dani88ge** Highly Voted 1 year, 10 months ago

age=number -> fifty two is not a number

upvoted 5 times

✉️  **gekkehenk** Most Recent 1 year, 4 months ago

Age = 52

Only number.

upvoted 1 times

Question #177

Topic 1

Where can Terraform not load a provider from?

- A. Source code
- B. Plugins directory
- C. Official HashiCorp distribution on releases.hashicorp.com
- D. Provider plugin cache

**Correct Answer: A**

*Community vote distribution*

A (74%)

C (21%)

5%

✉️  **amoyano** Highly Voted 1 year, 8 months ago

**Selected Answer: A**

- A- no info about TF loading a provider directly from source code, so this must be the correct answer.
- B- In the plugin directory are stored the providers, so TF could load providers from it
- C- releases.hashicorp.com is a valid source where providers can be downloaded from
- D- If the plugin provider cache is enabled eventually a provider could be loaded from it.

upvoted 11 times

✉️  **Pietjeplukgeluk** 1 year, 5 months ago

Totally agree on A. I do not see any information loading providers directly from source code.

upvoted 1 times

✉️  **ark007thegreat** (Most Recent) 6 months, 3 weeks ago

I did check the releases.hashicorp.com and found that terraform providers can be downloaded from that URL. For example you can refer the AWS provider from the URL '<https://releases.hashicorp.com/terraform-provider-aws/>'

upvoted 1 times

✉️  **piezas** 1 year, 1 month ago

**Selected Answer: C**

Terraform cannot load a provider from the Official HashiCorp distribution on releases.hashicorp.com.

The Official HashiCorp distribution on releases.hashicorp.com is the source where Terraform retrieves official provider releases. Terraform downloads provider binaries from this source and caches them in the provider plugin cache directory. The provider plugin cache is a location on the local machine where Terraform stores downloaded provider plugins to be used across multiple configurations.

upvoted 1 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer: A**

A. Source code.

Terraform can load a provider from a variety of sources, including the official HashiCorp distribution on releases.hashicorp.com, the provider plugin cache, and a plugins directory. However, Terraform cannot load a provider directly from the source code. Providers must be compiled and packaged as plugins before they can be loaded by Terraform. Terraform uses the provider's plugin binary to interact with the provider's API and manage the corresponding resources. Therefore, to use a provider in Terraform, it must be available as a compiled and packaged plugin in one of the supported locations.

upvoted 1 times

✉️  **Arrash** 1 year, 5 months ago

**Selected Answer: C**

releases.hashicorp.com is not related to terraform providers, its the main company site about all products

upvoted 1 times

✉️  **Abhijeet2904** 1 year, 5 months ago

**Selected Answer: B**

Terraform requires providers to be installed in its plugins directory. Providers obtained from other sources, such as source code or provider plugin caches, cannot be directly loaded by Terraform. Providers must be installed in the plugins directory to be properly recognized and used by Terraform.

upvoted 1 times

 **gsx** 1 year, 8 months ago

**Selected Answer: A**

When using the required\_providers setting to specify a version constraint, Terraform currently assumes the source is registry.terraform.io (or releases.hashicorp.com for older versions). Both registry.terraform.io and releases.hashicorp.com are populated by the providers grouped within the terraform-providers organization on GitHub.

upvoted 2 times

 **udibie** 1 year, 9 months ago

**Selected Answer: C**

C should be the answer

upvoted 2 times

 **jednorozec2022** 1 year, 9 months ago

C

The Releases API provides metadata about every HashiCorp product release hosted on our release site, HashiCorp Releases, and includes useful information such as links to release notes, licenses, associated repositories, and more.

ref: <https://www.hashicorp.com/blog/announcing-the-hashicorp-releases-api>

upvoted 3 times

 **kibkib** 1 year, 9 months ago

It seems to provide a provider for releases.hashicorp.com, but the answer is C though?

upvoted 1 times

 **jednorozec2022** 1 year, 9 months ago

<https://www.terraform.io/language/providers>

B/D is correct

no information about A, in my opinion incorrect is C

upvoted 1 times

Which of the following locations can Terraform use as a private source for modules? (Choose two.)

- A. Internally hosted SCM (Source Control Manager) platform
- B. Public Terraform Module Registry
- C. Private repository on GitHub
- D. Public repository on GitHub

**Correct Answer:** AC

*Community vote distribution*

AC (100%)

✉️  **Zeppoonstream** 1 year, 5 months ago

**Selected Answer:** AC

- A. Internally hosted SCM (Source Control Manager) platform
- C. Private repository on GitHub

Terraform can use the following locations as a private source for modules:

A. Internally hosted SCM (Source Control Manager) platform: Terraform allows you to specify a private SCM (Source Control Manager) platform like GitLab, Bitbucket, etc as a source for modules. This allows you to host your own modules within your organization's network, and use them privately.

C. Private repository on GitHub: Terraform allows you to specify a private repository on GitHub as a source for modules. This allows you to host your own modules on GitHub and use them privately.

upvoted 1 times

✉️  **Dcintel** 1 year, 10 months ago

**Selected Answer:** AC

AC for me

upvoted 2 times

✉️  **dani88ge** 1 year, 10 months ago

me too

upvoted 3 times

✉️  **keiffo2** 1 year, 10 months ago

I'd agree with A & C

upvoted 4 times

✉️  **dani88ge** 1 year, 10 months ago

me too

upvoted 1 times

Why should secrets not be hard coded into Terraform code? (Choose two.)

- A. It makes the code less reusable.
- B. Terraform code is typically stored in version control, as well as copied to the systems from which it's run. Any of those may not have robust security mechanisms.
- C. The Terraform code is copied to the target resources to be applied locally and could expose secrets if a target resource is compromised.
- D. All passwords should be rotated on a quarterly basis.

**Correct Answer: BC**

*Community vote distribution*

AB (57%)

BC (43%)

 **yaza** Highly Voted 1 year, 9 months ago

AB, terraform code will not be copied to target resource, if you deploy a VM the code will not be copied to the VM  
upvoted 18 times

 **yaza** 1 year, 9 months ago

Selected Answer: AB  
upvoted 3 times

 **3cc17f1** Most Recent 8 months, 3 weeks ago

I vote A and B, because C doesn't make sense. Why would the terraform code be copied to the target resources? For example, I provision an Azure Storage account using terraform. There's no point at which the terraform code ends up on that storage account.  
upvoted 2 times

 **brax404** 9 months ago

Selected Answer: AB

Explanation:

A. It makes the code less reusable: Hard coding secrets means the Terraform code is tied to a specific environment or set of credentials. This makes it hard to reuse the code in different contexts or environments without modifying the secrets.

B. Terraform code is typically stored in version control, as well as copied to the systems from which it's run. Any of those may not have robust security mechanisms: Storing secrets directly in the Terraform code exposes those secrets to anyone who has access to the code. Furthermore secrets may be logged in version control history, making them discoverable long after they've been removed or changed.  
upvoted 4 times

 **Aiwa23** 9 months ago

B and C. My Terraform source code is in github repo, and when I use pipelines to run terraform, the source code gets downloaded in the CI/CD or build server or terraform server and access holder to this server could see them. The question terms this server as the target resource. Yes, hardcoding does make it less reusable, but there is a way around- using environment specific tfvars.  
upvoted 1 times

 **vvkgp** 10 months, 4 weeks ago

Answer is B and C, as it's a serious security breach. A - just mentions about best practices.

upvoted 2 times

 **joyboy23** 1 year ago

Selected Answer: AB

AB, I don't think a terraform code is copied to any place(local, backend, any modules etc..) But, The values of the variables are rendered into state file. where the key/secrets are exposed  
upvoted 1 times

 **March2023** 1 year, 1 month ago

Selected Answer: BC

B and C  
upvoted 2 times

 **Rajmane** 11 months, 1 week ago

Exactly 100  
upvoted 1 times

👤 **kiran15789** 1 year, 2 months ago

**Selected Answer: BC**

B and C are security related

upvoted 2 times

👤 **FarziWaliMarzi** 1 year, 2 months ago

**Selected Answer: AB**

A and B

upvoted 1 times

👤 **Stanislav4907** 1 year, 3 months ago

**Selected Answer: BC**

B. Terraform code is typically stored in version control, as well as copied to the systems from which it's run. Any of those may not have robust security mechanisms. Storing secrets in plain text within code, especially if it's publicly accessible or shared, increases the risk of the secrets being compromised. If the code is stored in a version control system, it's important to ensure that the secrets are not accidentally exposed in version history.

C. The Terraform code is copied to the target resources to be applied locally and could expose secrets if a target resource is compromised. If Terraform code contains secrets, then those secrets will be copied to the target resources during the deployment process. If any of the target resources are compromised, the secrets may be exposed. It's important to keep secrets separate from the code and ensure that they are securely transmitted to the target resources when needed.

upvoted 1 times

👤 **joyboy23** 1 year ago

Will the code be copied though ?

upvoted 2 times

👤 **Chinensis** 1 year, 3 months ago

**Selected Answer: AB**

For me the answer C does not make sense...

upvoted 1 times

👤 **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: BC**

def, BC

upvoted 1 times

👤 **camps** 1 year, 3 months ago

**Selected Answer: BC**

B. Terraform code is typically stored in version control, as well as copied to the systems from which it's run. Any of those may not have robust security mechanisms.

C. The Terraform code is copied to the target resources to be applied locally and could expose secrets if a target resource is compromised.

Storing secrets, such as passwords or API keys, directly in Terraform code is a bad practice for several reasons. Firstly, Terraform code is typically stored in version control, and it may be copied to multiple systems from which it's run, such as a developer's machine, a CI/CD pipeline or a Terraform cloud workspace. Any of those systems may not have robust security mechanisms, and exposing secrets in code leaves them vulnerable to potential attacks. Secondly, the Terraform code is copied to the target resources to be applied locally, so any secrets in the code could be exposed if a target resource is compromised. Therefore, it is recommended to use a secrets management system, such as HashiCorp Vault or AWS Secrets Manager, to store and manage secrets outside of Terraform code.

upvoted 1 times

👤 **khaled\_razouk** 1 year, 4 months ago

**Selected Answer: BC**

B&C is the correct answer

upvoted 1 times

👤 **Daro\_** 1 year, 5 months ago

BC

in my opinion

upvoted 1 times

👤 **kounilasco** 1 year, 6 months ago

**Selected Answer: AB**

i choose A and B

upvoted 1 times

 **kounilasco** 1 year, 6 months ago

**Selected Answer: AB**

A and B are good answers  
upvoted 2 times

Question #180

Topic 1

If a Terraform creation-time provisioner fails, what will occur by default?

- A. The resource will not be affected, but the provisioner will need to be applied again
- B. The resource will be destroyed
- C. The resource will be marked as "tainted"
- D. Nothing, provisioners will not show errors in the command line

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **Burakko**  1 year, 10 months ago

**Selected Answer: C**

If a creation-time provisioner fails, the resource is marked as tainted. A tainted resource will be planned for destruction and recreation upon the next terraform apply .

upvoted 7 times

 **jednorozec2022** 1 year, 9 months ago

C

ref

<https://www.terraform.io/language/resources/provisioners/syntax#creation-time-provisioners>

upvoted 2 times

 **tabkar**  1 year, 2 months ago

**Selected Answer: C**

"If a creation-time provisioner fails, the resource is marked as tainted."

<https://developer.hashicorp.com/terraform/language/resources/provisioners/syntax>

upvoted 1 times

 **mamoon\_malta2022** 1 year, 4 months ago

Answer C:

If a creation-time provisioner fails, the resource is marked as tainted. A tainted resource will be planned for destruction and recreation upon the next terraform apply. Terraform does this because a failed provisioner can leave a resource in a semi-configured state. Because Terraform can reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

upvoted 3 times

Question #181

Topic 1

When should Terraform configuration files be written when running terraform import on existing infrastructure?

- A. Infrastructure can be imported without corresponding Terraform code
- B. Terraform will generate the corresponding configuration files for you
- C. You should write Terraform configuration files after the next terraform import is executed
- D. Terraform configuration should be written before terraform import is executed

**Correct Answer: B**

*Community vote distribution*

D (59%)

C (39%)

2%

✉️  **Uma10** Highly Voted 1 year, 10 months ago

**Selected Answer: D**

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version c Terraform will also generate configuration.

Because of this, prior to running terraform import it is necessary to write manually a resource configuration block for the resource, to which the imported object will be mapped.

Source: <https://www.terraform.io/cli/import>

upvoted 19 times

✉️  **zyxphreez** Highly Voted 1 year, 10 months ago

**Selected Answer: C**

i got confused..... check this url

<https://learn.hashicorp.com/tutorials/terraform/state-import>

the steps are

1. Identify the existing infrastructure you will import.
2. Import infrastructure into your Terraform state.
3. Write Terraform configuration that matches that infrastructure.
4. Review the Terraform plan to ensure the configuration matches the expected state and infrastructure.
5. Apply the configuration to update your Terraform state.

based on the learning path... i will choose C

upvoted 11 times

✉️  **SanderIsTheBestCloudShaper** 1 year, 10 months ago

this is true , its c

upvoted 1 times

✉️  **AB7088** 1 year, 6 months ago

the question said import on existing infrastructure, so it should be answer C. based on <https://learn.hashicorp.com/tutorials/terraform/state-import>

upvoted 1 times

✉️  **RVivek** 1 year, 9 months ago

2 second step import infrastructure into your Terraform state requires configuration to be written just an empty block with provider name and resource type and and resource address for example resource aws\_instance "my-webserver" { }

then you have to run terraform import ws\_instance .my-webserver <instance-id>

where instance-id of the Ec2 instance to be imported. Then the state is imported . from the state you have to copy other important values like VPC-id, subnet-id, ebs volume id ...

upvoted 3 times

✉️  **lezgino** 1 year, 5 months ago

Selected answer: D

That is correct, Terraform configuration should be written before terraform import is executed. It's best to have the Terraform code reflect the current state of the infrastructure so that Terraform can manage it.

upvoted 1 times

👤 **petersoliman** Most Recent 7 months, 2 weeks ago

**Selected Answer: C**

A. Infrastructure can be imported without corresponding Terraform code  
indeed but the question is saying should, so we should write the configurations sometime.

B. Terraform will generate the corresponding configuration files for you  
that is also correct but with the plan command not import  
terraform plan -generate-config-out=generated\_resources.tf  
that can be done after the importing, because you have the state file with the resources  
C. You should write Terraform configuration files after the next terraform import is executed  
that is the most correct for me.  
D. Terraform configuration should be written before terraform import is executed  
not sure why most people are choosing this one?!

upvoted 1 times

👤 **8876ca1** 1 month ago

Because of terraform import ADDR ID

upvoted 1 times

👤 **lukacs16** 1 year ago

**Selected Answer: D**

"To import a resource, first write a resource block for it in your configuration, establishing the name by which it will be known to Terraform"

Answer is D.

<https://developer.hashicorp.com/terraform/cli/import/usage>

upvoted 2 times

👤 **[Removed]** 1 year ago

**Selected Answer: C**

Will pick D:

Using configuration to import resources involves the following steps:

1. Identify the existing infrastructure you will import.
2. Define an import block for the resources.
3. Run terraform plan to review the import plan and optionally generate configuration for the resources.
4. Prune generated configuration to only the required arguments.
5. Apply the configuration to bring the resource into your Terraform state file.

<https://developer.hashicorp.com/terraform/tutorials/state/state-import>

upvoted 1 times

👤 **Jhaggar** 1 year, 2 months ago

**Selected Answer: D**

The correct statement is "Terraform configuration should be written before terraform import is executed". This is because Terraform import requires a corresponding resource block in the configuration file to identify and map the imported resource to the configuration. If the configuration file is not written before the import is executed, there will be no resource block available to map the imported resource to, which can result in an incomplete or incorrect state. After the import is executed, any necessary changes to the configuration file can be made to ensure that the configuration is up to date with the imported resources.

upvoted 1 times

👤 **FawadK** 1 year, 2 months ago

**Selected Answer: C**

Answer is C.

see <https://developer.hashicorp.com/terraform/tutorials/state/state-import>

upvoted 2 times

👤 **kiran15789** 1 year, 2 months ago

**Selected Answer: C**

the steps are

1. Identify the existing infrastructure you will import.
2. Import infrastructure into your Terraform state.
3. Write Terraform configuration that matches that infrastructure.
4. Review the Terraform plan to ensure the configuration matches the expected state and infrastructure.
5. Apply the configuration to update your Terraform state.

upvoted 2 times

👤 **Chinensis** 1 year, 3 months ago

**Selected Answer: C**

The answer is clearly written here:

<https://developer.hashicorp.com/terraform/tutorials/state/state-import>

Why debate? :)

upvoted 3 times

👤 **camps** 1 year, 3 months ago

**Selected Answer: D**

D. Terraform configuration should be written before terraform import is executed

When running terraform import on existing infrastructure, you should write the corresponding Terraform configuration files before executing the import command. The terraform import command requires that the resource configuration already exists in your Terraform code, as it maps the existing infrastructure to the defined resource in your configuration. The command then adds the imported resource information to your Terraform state file.

upvoted 1 times

👤 **camps** 1 year, 3 months ago

**Selected Answer: D**

D. Terraform configuration should be written before terraform import is executed.

Terraform import is used to import an existing infrastructure into a Terraform configuration. Before running terraform import, the configuration must be written to describe the desired state of the imported resources. Once the configuration is written and the import command is executed, Terraform maps the existing resources to the configuration and updates its state accordingly. Therefore, Terraform configuration files should be written before running terraform import on existing infrastructure.

upvoted 1 times

👤 **noobster** 1 year, 3 months ago

Before you run terraform import you must manually write a resource configuration block for the resource.

<https://developer.hashicorp.com/terraform/cli/import>

upvoted 1 times

👤 **TechHero** 1 year, 5 months ago

You don't need to write resource configuration before importing

Bringing existing infrastructure under Terraform's control involves five steps:

1. Identify the existing infrastructure you will import.
2. Import infrastructure into your Terraform state.
3. Write Terraform configuration that matches that infrastructure.
4. Review the Terraform plan to ensure the configuration matches the expected state and infrastructure.
5. Apply the configuration to update your Terraform state.

[https://developer.hashicorp.com/terraform/tutorials/state/state-import?utm\\_source=WEBSITE&utm\\_medium=WEB\\_IO&utm\\_offer=ARTICLE\\_PAGE&utm\\_content=DOCS](https://developer.hashicorp.com/terraform/tutorials/state/state-import?utm_source=WEBSITE&utm_medium=WEB_IO&utm_offer=ARTICLE_PAGE&utm_content=DOCS)

The Correct Answer is: A

upvoted 2 times

👤 **Zeppoonstream** 1 year, 5 months ago

**Selected Answer: D**

D. Terraform configuration should be written before terraform import is executed

When running the terraform import command on existing infrastructure, Terraform uses the configuration files to know how to interact with the resources. Without the configuration files, Terraform does not know how to interact with the resources and what to import. Therefore, Terraform configuration files should be written before terraform import is executed.

- A. Infrastructure can be imported without corresponding Terraform code is not true.
- B. Terraform will generate the corresponding configuration files for you is not true.

C. You should write Terraform configuration files after the next terraform import is executed is not true.

It's important to note that Terraform import is not a replacement for creating Terraform configuration files, but it helps to discover and populate the state with existing resources.

upvoted 1 times

 **gekkehenk** 1 year, 6 months ago

**Selected Answer: D**

According to Terraform, the configuration should be written before performing the import.

"Before you run terraform import you must manually write a resource configuration block for the resource. The resource block describes where Terraform should map the imported object."

<https://developer.hashicorp.com/terraform/cli/import>

upvoted 1 times

 **Andreadw** 1 year, 6 months ago

**Selected Answer: A**

to import is not necessary update config. To apply yes, is tricky

upvoted 1 times

 **robertninho** 1 year, 6 months ago

D. Terraform configuration should be written before terraform import is executed

The terraform import command is used to import existing infrastructure into Terraform, so that it can be managed as part of your configuration. In order to use terraform import, you need to have a corresponding Terraform configuration file that defines the desired state of the infrastructure.

Therefore, it is generally a good idea to write the Terraform configuration file before running terraform import. This will allow you to specify the desired state of the infrastructure, including any resources that you want to import and any additional resources or configurations that you want to add.

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: D**

Terraform configuration files should be written before running the terraform import command on existing infrastructure. The configuration files define the infrastructure that Terraform will manage, and the terraform import command is used to import existing resources into the configuration so that Terraform can manage them.

upvoted 1 times

Which command lets you experiment with Terraform's built-in functions?

- A. terraform env
- B. terraform console
- C. terraform test
- D. terraform validate

**Correct Answer: B**

Reference:

<https://www.terraform.io/language/functions>

*Community vote distribution*

B (100%)

✉️  BaburTurk 10 months, 3 weeks ago

**Selected Answer: B**

The terraform console command lets you experiment with Terraform's built-in functions. It provides an interactive shell where you can type in Terraform expressions and see the results.

The other commands are used for different purposes:

terraform env lists the available Terraform environments.  
terraform test runs unit tests on your Terraform configuration.  
terraform validate checks your Terraform configuration for errors.

upvoted 1 times

✉️  phidelics 1 year, 3 months ago

Terraform Console

upvoted 1 times

✉️  X\_Wuo 1 year, 4 months ago

**Selected Answer: B**

The provided answer is correct.

upvoted 1 times

✉️  gekkehenk 1 year, 4 months ago

**Selected Answer: B**

Terraform console

This command provides an interactive command-line console for evaluating and experimenting with expressions. You can use it to test interpolations before using them in configurations and to interact with any values currently saved in state.

<https://developer.hashicorp.com/terraform/cli/commands/console>

upvoted 1 times

✉️  depal\_dhir 1 year, 10 months ago

**Selected Answer: B**

<https://www.terraform.io/cli/commands/console>

upvoted 1 times

✉️  Uma10 1 year, 10 months ago

The provided answer is correct.

Terraform console provides an interactive command-line console for evaluating and experimenting with expressions. You can use it to test interpolations before using them in configurations and to interact with any values currently saved in state.

Source: <https://www.terraform.io/cli/commands/console>

upvoted 3 times

Question #183

Topic 1

Why does this backend configuration not follow best practices?

```
terraform {
 backend "s3" {
 bucket = "terraform-state-prod"
 key = "network/terraform.tfstate"
 region = "us-east-1"
 access_key = "AKIAIOSFODNN7EXAMPLE"
 secret_key = "wJalrXUtnFEMI/K7MDENG/bPxRfICYEXAMPLEKEY"
 }

 required_providers {
 aws = {
 source = "hashicorp/aws"
 version = "~> 3.38"
 }
 }

 required_version = ">= 0.15"
}
```

- A. You should not store credentials in Terraform Configuration

- B. You should use the local enhanced storage backend whenever possible
- C. An alias meta-argument should be included in backend blocks whenever possible
- D. The backend configuration should contain multiple credentials so that more than one user can execute terraform plan and terraform apply

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: A**

%100 A

upvoted 7 times

 **dепал\_dхир** Highly Voted 1 year, 10 months ago

**Selected Answer: A**

Access Key and Secret Key inside the config file? Really?

upvoted 5 times

 **Alandt** Most Recent 5 months, 3 weeks ago

**Selected Answer: A**

Easy come on, everyone should know this.

upvoted 1 times

 **phidelics** 1 year, 3 months ago

**Selected Answer: A**

Never and never store your credentials in your terraform config

upvoted 1 times

 **kennynelcon** 1 year, 5 months ago

**Selected Answer: A**

A is the answer

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

Nobrainer

upvoted 3 times

Question #184

Topic 1

Open source Terraform can only import publicly-accessible and open-source modules.

A. True

B. False

**Correct Answer: A**

### Community vote distribution

B (82%)

A (18%)

✉️  **Uma10** Highly Voted 1 year, 10 months ago

**Selected Answer: B**

Terraform can load modules from a public or private registry. This makes it possible to publish modules for others to use, and to use modules others have published.

Also, members of your organization might produce modules specifically crafted for your own infrastructure needs. Terraform Cloud and Terraform Enterprise both include a private module registry for sharing modules internally within your organization.

Source: <https://www.terraform.io/language/modules>

upvoted 10 times

✉️  **TafMuko** Most Recent 12 months ago

**Selected Answer: A**

<https://www.hashicorp.com/resources/why-consider-terraform-enterprise-over-open-source>

upvoted 1 times

✉️  **Jhaggar** 1 year, 2 months ago

**Selected Answer: B**

False as it can be from anywhere

upvoted 1 times

✉️  **pyro7** 1 year, 4 months ago

The Correct answer is B

upvoted 1 times

✉️  **princajen** 1 year, 4 months ago

**Selected Answer: B**

B. False.

Open source Terraform can import both publicly-accessible and private modules from various sources, including Terraform Registry, GitHub, GitLab, Bitbucket, and others. However, the support for private modules is limited in the open source version, while it's fully supported in Terraform Enterprise.

upvoted 2 times

✉️  **wanrltw** 1 year, 7 months ago

**Selected Answer: A**

It's A! Don't confuse Terraform Open Source with Cloud/Enterprise.

<https://developer.hashicorp.com/terraform/intro/terraform-editions>

<https://www.hashicorp.com/resources/why-consider-terraform-enterprise-over-open-source>

upvoted 2 times

✉️  **robertninho** 1 year, 6 months ago

```
module "suvpc" {
 source = "git::https://user:password@github.com/example/vpc.git"
}
```

upvoted 3 times

✉️  **waldonuts** 1 year, 10 months ago

**Selected Answer: B**

Private repo for custom modules.

upvoted 1 times

 **duffyduck** 1 year, 10 months ago

Published Modules

In addition to modules from the local filesystem, Terraform can load modules from a public or private registry. This makes it possible to publish modules for others to use, and to use modules that others have published.

The Terraform Registry hosts a broad collection of publicly available Terraform modules for configuring many kinds of common infrastructure. These modules are free to use, and Terraform can download them automatically if you specify the appropriate source and version in a module call block.

Also, members of your organization might produce modules specifically crafted for your own infrastructure needs. Terraform Cloud and Terraform Enterprise both include a private module registry for sharing modules internally within your organization.

Answer is B

upvoted 3 times

Question #185

Topic 1

What does terraform destroy do?

- A. Destroy all infrastructure in the Terraform state file
- B. Destroy all Terraform code files in the current directory while leaving the state file intact
- C. Destroy all infrastructure in the configured Terraform provider
- D. Destroy the Terraform state file while leaving infrastructure intact

**Correct Answer: D**

*Community vote distribution*

A (83%)

C (17%)

 **dinesh198728**  1 year, 10 months ago

**Selected Answer: A**

The terraform destroy command terminates resources managed by your Terraform project. This command is the inverse of terraform apply in that it terminates all the resources specified in your Terraform state. It does not destroy resources running elsewhere that are not managed by the current Terraform project.

<https://learn.hashicorp.com/tutorials/terraform/aws-destroy>

upvoted 15 times

 **kiran15789**  1 year, 2 months ago

**Selected Answer: A**

Option C is not correct because the command only destroys the infrastructure defined in the Terraform configuration files, not all infrastructure in the configured Terraform provider.

upvoted 2 times

 **Nunyabiznes** 1 year, 3 months ago

answers are : AC

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: A**

A. Destroy all infrastructure in the Terraform state file.

Terraform destroy is a command used to destroy all the infrastructure resources that are defined in the Terraform configuration file. This command removes all resources created by the Terraform configuration and deletes them from the infrastructure provider, effectively tearing down the entire infrastructure that was created using Terraform. The command operates on the Terraform state file, which is used to track the current state of the infrastructure. When the command is executed, it removes all resources in the state file and makes the infrastructure match the desired state of having no resources.

upvoted 1 times

 **phidelics** 1 year, 3 months ago

**Selected Answer: A**

destroy all infrastructure

upvoted 1 times

 **Zeppoonstream** 1 year, 5 months ago

**Selected Answer: A**

A. Destroy all infrastructure in the Terraform state file

terraform destroy command is used to destroy all infrastructure resources that are managed by Terraform in the current state file. This command will ask for confirmation before proceeding with the destruction and will remove the resources from the provider, and also remove the resource from the state file.

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

A - is correct

upvoted 1 times

 **DerekKey** 1 year, 7 months ago

A - Correct

Terraform destroy destroys every resource that was created by Terraform - it means every resource stored in terraform.tfstate

upvoted 3 times

 **G4Exams** 1 year, 8 months ago

**Selected Answer: A**

A is right. The rest are distractors

upvoted 1 times

 **dk\_2022** 1 year, 8 months ago

C is the right answer:

The terraform destroy command destroys all of the resources being managed by the current working directory and workspace, using state data to determine which real world objects correspond to managed resources.

A destroy behaves exactly like deleting every resource from the configuration

upvoted 2 times

 **alifie** 1 year, 9 months ago

**Selected Answer: A**

A it is

upvoted 1 times

 **lordogre16** 1 year, 9 months ago

**Selected Answer: C**

It's C

"The terraform destroy command is used to destroy the project infrastructure and free up the allocated resources. terraform destroy only destroys the infrastructure for the current project. There are no changes made to the resources allocations running elsewhere."

<https://www.educative.io/answers/what-is-the-command-to-destroy-infrastructure-in-terraform>

upvoted 1 times

 **lordogre16** 1 year, 9 months ago

I take this all back it's actually A, it modifies the state file not the set provider

upvoted 1 times

 **lezhino** 1 year, 5 months ago

Destroy all infrastructure in the Terraform state file is the correct answer. The terraform destroy command is used to destroy the infrastructure that has been created and managed by Terraform, as described in the Terraform documentation:

<https://www.terraform.io/docs/commands/destroy.html>.

A is correct]

upvoted 1 times

 **duffyduck** 1 year, 10 months ago

This is a question about Terraform -Destory

Command: destroy

The terraform destroy command is a convenient way to destroy all remote objects managed by a particular Terraform configuration.

While you will typically not want to destroy long-lived objects in a production environment, Terraform is sometimes used to manage ephemeral infrastructure for development purposes, in which case you can use terraform destroy to conveniently clean up all of those temporary objects once you are finished with your work.

upvoted 2 times

 **shopkitty** 1 year, 10 months ago

**Selected Answer: A**

I votes for A

upvoted 1 times

 **geekneek** 1 year, 10 months ago

**Selected Answer: C**

I think is C.

"The terraform destroy command terminates resources managed by your Terraform project. This command is the inverse of terraform apply in that it terminates all the resources specified in your Terraform state. It does not destroy resources running elsewhere that are not managed by current Terraform project."

<https://learn.hashicorp.com/tutorials/terraform/aws-destroy>

upvoted 1 times

 **dinesh198728** 1 year, 10 months ago

**Selected Answer: A**

Must be A

upvoted 3 times

 **donathon** 1 year, 10 months ago

**Selected Answer: A**

I vote for A. The destroy will only delete those that are in the state. Eg. even if you have a new resource configured after the last apply (which would be in the state), it will not delete this resources as it cannot be found in the state.

upvoted 2 times

While attempting to deploy resources into your cloud provider using Terraform, you begin to see some odd behavior and experience sluggish responses. In order to troubleshoot you decide to turn on Terraform debugging. Which environment variables must be configured to make Terraform's logging more verbose?

- A. TF\_LOG\_LEVEL
- B. TF\_LOG\_FILE
- C. TF\_LOG
- D. TP\_LOG\_PATH

**Correct Answer: C**

Reference:

<https://www.terraform.io/internals/debugging>

*Community vote distribution*

C (100%)

 **secdaddy** 1 year, 7 months ago

The question asks for variables plural but configuring a log path to have a file does not make logging more verbose, so only C.  
upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: C**

<https://www.terraform.io/internals/debugging>

upvoted 2 times

 **keiffo2** 1 year, 10 months ago

You can set TF\_LOG to one of the log levels TRACE, DEBUG, INFO, WARN or ERROR to change the verbosity of the logs. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

Type "TF\_LOG=TRACE TF\_LOG\_PATH=./ terraform apply"

upvoted 1 times

 **keiffo2** 1 year, 10 months ago

Answer C

upvoted 1 times

If a DevOps team adopts AWS CloudFormation as their standardized method for provisioning public cloud resources, which of the following scenarios poses a challenge for this team?

- A. The team is asked to build a reusable code base that can deploy resources into any AWS region
- B. The team is asked to manage a new application stack built on AWS-native services
- C. The organization decides to expand into Azure and wishes to deploy new infrastructure using their existing codebase
- D. The DevOps team is tasked with automating a manual provisioning process

**Correct Answer: D**

*Community vote distribution*

C (100%)

 **camps** 1 year, 3 months ago

**Selected Answer: C**

C. The organization decides to expand into Azure and wishes to deploy new infrastructure using their existing codebase

AWS CloudFormation is a service specifically designed for provisioning AWS resources. If the organization decides to expand into Azure and deploy new infrastructure using the existing codebase, the DevOps team will face a challenge, as CloudFormation does not support Azure or other cloud providers.

upvoted 1 times

 **wanrltw** 1 year, 7 months ago

**Selected Answer: C**

<https://developer.hashicorp.com/terraform/intro/vs/cloudformation>

upvoted 2 times

 **shopkitty** 1 year, 10 months ago

**Selected Answer: C**

Will have trouble if want to deploy new resource in Azure as AWS CloudFormation only for AWS.

upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: C**

Cloudformation is for AWS only

upvoted 1 times

 **Burakko** 1 year, 10 months ago

**Selected Answer: C**

It is definitely C. With cloudformation, Azure infrastructure cannot be built.

upvoted 4 times

 **zyxphreez** 1 year, 10 months ago

Should be C..... cloudformation is only for aws..

upvoted 4 times

You cannot install third party plugins using terraform init.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: B**

In June at HashiConf digital we announced the beta version of HashiCorp Terraform 0.13. Many of the improvements in Terraform 0.13 focus on the diverse, rapidly-growing collection of official, partner, and community providers. With Terraform 0.13, terraform init will automatically download and install partner and community providers in the HashiCorp Terraform Registry, following the same clear workflow as HashiCorp-supported official providers. These improvements to the ecosystem will benefit Terraform users and provider developers alike.

upvoted 6 times

 **camps** Most Recent 1 year, 3 months ago

**Selected Answer: B**

B. False

Terraform does support installing third-party plugins, but the process is not as straightforward as it is for official providers. Starting with Terraform 0.13, you can install third-party providers using the terraform init command if you have properly configured the required\_provider block and set up the provider source.

upvoted 1 times

 **depal\_dhir** 1 year, 10 months ago

**Selected Answer: B**

<https://www.terraform.io/cli/commands/init>

upvoted 2 times

 **Uma10** 1 year, 10 months ago

**Selected Answer: B**

The provided answer is correct.

For providers that are published in either the public Terraform Registry or in a third-party provider registry, terraform init will automatically find, download, and install the necessary provider plugins.

Source: <https://www.terraform.io/cli/commands/init>

upvoted 4 times

Which of the following can you do with terraform plan? (Choose two.)

- A. Save a generated execution plan to apply later
- B. Execute a plan in a different workspace
- C. View the execution plan and check if the changes match your expectations
- D. Schedule Terraform to run at a planned time in the future

**Correct Answer:** AC

Reference:

<https://learn.hashicorp.com/tutorials/terraform/plan>

*Community vote distribution*

AC (100%)

 depal\_dhir Highly Voted 1 year, 10 months ago

**Selected Answer:** AC

<https://learn.hashicorp.com/tutorials/terraform/plan>

upvoted 5 times

 camps Most Recent 1 year, 3 months ago

**Selected Answer:** AC

- A. Save a generated execution plan to apply later
- C. View the execution plan and check if the changes match your expectations

upvoted 1 times

 G4Exams 1 year, 8 months ago

**Selected Answer:** AC

C is the main purpose of plan and A can be done with plan -out=FILE flag. So A & C.

upvoted 2 times

Which are examples of infrastructure as code? (Choose two.)

- A. Cloned virtual machine images
- B. Change management database records
- C. Versioned configuration files
- D. Docker files

**Correct Answer:** BC

*Community vote distribution*

CD (100%)

 **Burakko** Highly Voted 1 year, 10 months ago

**Selected Answer: CD**

To me it is more like CD.  
upvoted 5 times

 **Zeppoonstream** Highly Voted 1 year, 5 months ago

**Selected Answer: CD**

C. Versioned configuration files  
D. Docker files

Examples of infrastructure as code are:

C. Versioned configuration files: Infrastructure as code is when infrastructure is defined and managed using code, and configuration files that written in a programming language, such as JSON or YAML, and stored in a version control system like Git, allowing for versioning and collaboration.

D. Docker files: Docker files are a type of infrastructure as code that describe how to build a container image. They are written in a simple programming language and can be versioned and stored in a version control system.

upvoted 5 times

 **Chinensis** Most Recent 1 year, 3 months ago

IAC also means working with a change management tool. So I hesitate between (B or D) and C.

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: CD**

C. Versioned configuration files  
D. Docker files

Infrastructure as code (IAC) refers to the practice of managing and provisioning infrastructure through code, rather than manual processes. IA enables version control, collaboration, and automation for infrastructure management.

upvoted 1 times

 **G4Exams** 1 year, 8 months ago

**Selected Answer: CD**

I go for C and D but A and D also possible...not 100% sure  
upvoted 2 times

 **RVivek** 1 year, 9 months ago

**Selected Answer: CD**

How a cloned image or DB record can be considered as IaC. It does not make sense to me.  
C& D are the answer  
upvoted 2 times

 **dепал\_dхир** 1 year, 10 months ago

**Selected Answer: CD**

Version Config File - Git

Question #191

Topic 1

FILL BLANK -

You need to migrate a workspace to use a remote backend. After updating your configuration, what command do you run to perform the migration?

Type your answer in the field provided. The text field is not case-sensitive and all variations of the correct answer are accepted.

**Correct Answer: terraform init**

 **dinesh198728** Highly Voted 1 year, 10 months ago

terraform init

Once you have authenticated to Terraform Cloud, you're ready to migrate your local state file to Terraform Cloud. To begin the migration, reinitialize. This causes Terraform to recognize your cloud block configuration.

upvoted 8 times

⊕  **XP\_2600**  1 year, 5 months ago

I think terraform init -migrate-state

The -migrate-state option will attempt to copy existing state to the new backend, and depending on what changed, may result in interactive prompts to confirm migration of workspace states. The -force-copy option suppresses these prompts and answers "yes" to the migration questions. Enabling -force-copy also automatically enables the -migrate-state option.

<https://developer.hashicorp.com/terraform/cli/commands/init>

upvoted 7 times

⊕  **tuso**  1 month, 1 week ago

Just did it some days ago (from local to s3). I did "terraform init" and then terraform asks to confirm the migration of your state file. No "-migrate-state" needed.

upvoted 1 times

⊕  **Komas1999** 1 month, 2 weeks ago

terraform init -migrate-state missing the last part in the correct answer

upvoted 1 times

⊕  **alen995454** 6 months ago

Re-running init with an already-initialized backend will update the working directory to use the new backend settings. Either -reconfigure or -migrate-state must be supplied to update the backend configuration

<https://developer.hashicorp.com/terraform/cli/commands/init>

I'm going w/ : terraform init -migrate-state

upvoted 2 times

⊕  **BaburTurk** 10 months, 3 weeks ago

terraform init -migrate-state.

This command will tell Terraform to read the existing state file and migrate it to the new remote backend.

The -migrate-state flag is only available in Terraform 0.15 and later. If you are using an older version of Terraform, you will need to use the terraform state mv

command to manually migrate the state file.

Here is an example of how to use the terraform init -migrate-state command:

```
terraform init -migrate-state \
-backend=s3://my-bucket/my-workspace
```

upvoted 2 times

⊕  **Nick\_001** 1 year, 4 months ago

terraform init -migrate-state

upvoted 4 times

⊕  **LeyLey** 1 year, 5 months ago

terraform init --reconfigure is correct

upvoted 1 times

⊕  **Daro\_** 1 year, 5 months ago

terraform init --reconfigure should be correct<<<<<<<<

<https://support.hashicorp.com/hc/en-us/articles/360001151948-Migrate-Workspace-State-Using-Terraform-State-Push-Pull>

upvoted 2 times

⊕  **yair319732** 1 year, 5 months ago

terraform init --reconfigure , but I think terraform init will be sufficient.

upvoted 1 times

✉️  **fedeX** 1 year, 6 months ago

terraform init -upgrade

To migrate a workspace to use a remote backend, you need to run the terraform init command with the -upgrade flag. This will cause Terraform to upgrade the workspace to use the new backend configuration.

For example, if you have updated your configuration to use an S3 backend.

This will cause Terraform to create any necessary resources in the remote backend (such as an S3 bucket) and migrate the workspace state to the new backend. It is important to note that this will overwrite any existing state in the remote backend, so it is recommended to take a backup of the state before running this command.

upvoted 1 times

✉️  **E\_awS** 1 year, 6 months ago

"terraform init" would be enough

upvoted 2 times

✉️  **Daro\_** 1 year, 5 months ago

terraform init -upgrade -- to upgrade version of providers which are already installed

upvoted 1 times

✉️  **BaburTurk** 1 year, 6 months ago

<https://support.hashicorp.com/hc/en-us/articles/360001151948-Migrate-Workspace-State-Using-Terraform-State-Push-Pull>

upvoted 1 times

Question #192

Topic 1

When using a module from the public Terraform Module Registry, the following parameters are required attributes in the module block. (Choose two.)

- A. Each of the module's required inputs
- B. The module's source address
- C. Terraform Module Registry account token
- D. Each of the module's dependencies (example: submodules)
- E. The version of the module

**Correct Answer: BE**

*Community vote distribution*

AB (57%)

BE (40%)

3%

✉️  **kareem\_ashraf** 7 months, 3 weeks ago

**Selected Answer: BE**

Modules on the public Terraform Registry can be referenced using a registry source address of the form <NAMESPACE>/<NAME>/<PROVIDER> with each module's information page on the registry site including the exact address to use.

```
module "consul" {
 source = "hashicorp/consul/aws"
 version = "0.1.0"
}
```

upvoted 2 times

✉️  **kareem\_ashraf** 7 months, 3 weeks ago

I think iam wrong version is not needed I would go for A,B

upvoted 3 times

👤 **petersoliman** 7 months, 3 weeks ago

Selected Answer: AB

A and B

upvoted 2 times

👤 **Jas14** 9 months, 3 weeks ago

The version argument is not required, but we highly recommend you include it when using a Terraform module. For supported sources, this argument specifies the module version Terraform will load. Without the version argument, Terraform will load the latest version of the module. Modules can contain both required and optional arguments. You must specify all required arguments to use the module.

from Hashicorp's official site.

upvoted 1 times

👤 **seifskl** 12 months ago

Selected Answer: AB

A. Each of the module's required inputs

B. The module's source address

These two parameters are required when using a module from the Terraform Module Registry. The module's source address tells Terraform where to find the module. The required inputs for the module must be specified so that Terraform can configure the module correctly.

The Terraform Module Registry does not require an account token for use, and dependencies should be managed inside the module itself. The version of the module is not strictly required; if not provided, Terraform will use the latest version available.

You can refer to this link <https://developer.hashicorp.com/terraform/language/modules/syntax>

it is mentioned that :

- "The source argument is mandatory for all modules."
- "All modules require a source argument"
- "The version argument is recommended for modules from a registry." which means that it is not required

upvoted 4 times

👤 **wh1t4k3r** 12 months ago

Selected Answer: AB

Version is not required

<https://developer.hashicorp.com/terraform/language/expressions/version-constraints>

upvoted 3 times

👤 **Talamak** 1 year ago

A little bit of confusion here, so I've just tested it and called a module from public registry, specifying the source and required variables only as worked. Therefore, I suggest the correct answer is A, B.

upvoted 1 times

👤 **raf123123** 1 year ago

Selected Answer: BE

Answer BE

upvoted 1 times

👤 **vibzr2023** 3 months, 1 week ago

E Version is not required

upvoted 1 times

👤 **[Removed]** 1 year ago

Selected Answer: BE

Look under the Provision Instructions. Only source and version are required.

<https://registry.terraform.io/modules/terraform-aws-modules/vpc/aws/latest>

upvoted 2 times

👤 **Ni33** 1 year, 2 months ago

Selected Answer: BE

B and E are correct answers. A is not correct option because input variables can be defaulted to.

upvoted 3 times

👤 Stanislav4907 1 year, 3 months ago

Selected Answer: BE

B. The module's source address and E. The version of the module are required attributes in the module block when using a module from the public Terraform Module Registry.

The source parameter specifies the source location of the module, which can be a Git URL, local file path, or a Terraform Registry module identifier. This is necessary to locate the module to be used.

The version parameter specifies the version constraint of the module, which can be a specific version number or a version constraint. This is necessary to ensure that a consistent version of the module is used across different Terraform runs.

The other options listed are not required attributes in the module block. Each of the module's required inputs are defined separately in the module block and can be specified in the module block as needed. The Terraform Module Registry account token is not required to use a public module. The module's dependencies (example: submodules) are not required in the module block but can be defined in the module as necessary.

upvoted 2 times

👤 camps 1 year, 3 months ago

Selected Answer: AB

When using a module from the public Terraform Module Registry, you need to include certain attributes in the module block to properly reference and configure the module.

A) Each of the module's required inputs:

Modules can have input variables that allow users to customize the behavior of the module. In the module block, you must provide values for required input variables. If a module has optional input variables, you can also provide values for those, but they will have default values if not specified.

B) The module's source address:

The source address tells Terraform where to find the module's source code. For modules hosted on the Terraform Module Registry, the source address is usually in the format <NAMESPACE>/<NAME>/<PROVIDER>.

The version of the module is an optional attribute. If not specified, Terraform will use the most recent version of the module that is compatible with your Terraform version. However, it is a good practice to specify a version to ensure consistent behavior across different environments and Terraform runs.

upvoted 3 times

👤 mamoon\_malta2022 1 year, 4 months ago

Answer is A & B

version is not a required attribute, if you do not specify version it will get the latest version by default

upvoted 2 times

👤 Zeppoonstream 1 year, 5 months ago

A. Each of the module's required inputs: When using a Terraform module, you need to specify the input variables the module requires. This ensures the module has all the necessary information to run correctly. You need to provide values for each of the module's required inputs in the module block.

B. The module's source address: The source address specifies where Terraform should find the module in the Terraform Module Registry. This includes the registry name and the module name, along with the version of the module you want to use. The source address is a required attribute in the module block and it helps Terraform download and install the module from the registry.

upvoted 1 times

👤 Only5 1 year, 5 months ago

I go with B and E w.r.t

The source argument is mandatory for all modules.

The version argument is recommended for modules from a registry.

<https://developer.hashicorp.com/terraform/language/modules/syntax>

upvoted 1 times

👤 Nzudin 1 year, 4 months ago

if there is no version set it will take the latest one so it is not required.

upvoted 1 times

👤 yair319732 1 year, 5 months ago

Selected Answer: AB

The answer is A,B .. version isn't mandatory:

" "

Registry modules support versioning. You can provide a specific version as shown in the above examples, or use flexible version constraints.

" "

upvoted 1 times

 **Zeppoonstream** 1 year, 5 months ago

**Selected Answer: BE**

- B. The module's source address
- E. The version of the module

When using a module from the public Terraform Module Registry, the following parameters are required attributes in the module block:

B. The module's source address: The source argument in the module block is required and it contains the address of the module in the public Terraform Module Registry.

E. The version of the module: The version of the module is also required and it is used to specify which version of the module should be used.

A. Each of the module's required inputs: The required inputs are defined by the module provider and should be passed as variables to the module block.

C. Terraform Module Registry account token: Account token is not required, it could be used for authentication but it's not required for using a module from the public Terraform Module Registry.

D. Each of the module's dependencies (example: submodules): Dependencies are not required, but if the module has any submodules, it should be added as a source of the submodule in the root module.

upvoted 2 times

 **gekkehenk** 1 year, 6 months ago

**Selected Answer: AB**

Version is not mandatory. Only mandatory is "source" and the input vars that are mandatory for the module to operate.

upvoted 4 times

Question #193

Topic 1

As a developer, you want to ensure your plugins are up to date with the latest versions. Which Terraform command should you use?

- A. terraform init -upgrade
- B. terraform apply -upgrade
- C. terraform refresh -upgrade
- D. terraform providers -upgrade

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Abuu** 1 year, 5 months ago

A is the answer

upvoted 3 times

 **Zeppoonstream** 1 year, 5 months ago

duplicate

upvoted 1 times

 **sribalaje** 1 year, 5 months ago

**Selected Answer: A**

A is correct

upvoted 2 times

You can access state stored with the local backend by using the `terraform_remote_state` data source.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

A (69%)

B (31%)

✉  Stanislav4907  1 year, 3 months ago

**Selected Answer: A**

The `terraform_remote_state` data source allows you to retrieve outputs from the state of another Terraform configuration, which can be stored local or remote backend. To use the `terraform_remote_state` data source with a local backend, you would define the path to the state file in the backend configuration block of your Terraform configuration, and then use that path in the data "terrafrom\_remote\_state" block to retrieve the desired output values.

upvoted 8 times

✉  VSMu  12 months ago

**Selected Answer: A**

Example <https://developer.hashicorp.com/terraform/language/state/remote-state-data#example-usage-local-backend>  
upvoted 2 times

✉  FarziWaliMarzi 1 year, 3 months ago

**Selected Answer: A**

<https://developer.hashicorp.com/terraform/language/state/remote-state-data>

```
data "terraform_remote_state" "vpc" {
 backend = "local"
```

```
 config = {
 path = "..."
 }
}
```

```
Terraform >= 0.12
resource "aws_instance" "foo" {
...
 subnet_id = data.terraform_remote_state.vpc.outputs.subnet_id
}
```

```
Terraform <= 0.11
resource "aws_instance" "foo" {
...
 subnet_id = "${data.terraform_remote_state.vpc.subnet_id}"
}
```

upvoted 4 times

✉  Nunyabiznes 1 year, 3 months ago

**Selected Answer: B**

The `terraform_remote_state` data source is used to access state data stored in a remote backend (such as S3, GCS, or Terraform Cloud). It is designed to access local backend state files directly. However, if you want to access data from another Terraform configuration's local state, you can use output variables and pass the values between configurations using input variables or other methods.

upvoted 2 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

The `terraform_remote_state` data source is used to access the state data from another Terraform configuration that is stored in a remote backend, not a local backend. This data source enables you to fetch the outputs of another Terraform project and use them as input variables for your current Terraform project.

When using the local backend, the Terraform state is stored in a local file (usually named `terraform.tfstate`). This means the state is not accessed through the `terraform_remote_state` data source, as it is not stored remotely.

upvoted 1 times

 **SaadKhamis** 1 year, 3 months ago

It is possible to use it to access local backend.

<https://developer.hashicorp.com/terraform/language/state/remote-state-data#example-usage-local-backend>

```
data "terraform_remote_state" "vpc" {
 backend = "local"
```

```
 config = {
 path = "..."
 }
}
```

upvoted 2 times

 **Strib** 1 year, 3 months ago

**Selected Answer: A**

Example Usage (local Backend)

<https://developer.hashicorp.com/terraform/language/state/remote-state-data#example-usage-local-backend>

upvoted 3 times

 **David\_C\_90** 1 year, 4 months ago

**Selected Answer: A**

I tested this and I believe this is correct. Check the link <https://developer.hashicorp.com/terraform/language/state/remote-state-data#example-usage-local-backend>

Note that the state file must already exist and the data you will obtain is the data present on the file: if you change the terraform configuration corresponding to the state you are trying to read, you will obtain the "old" data.

upvoted 1 times

 **khaled\_razouk** 1 year, 4 months ago

**Selected Answer: B**

I'll go with B

upvoted 1 times

 **rotimislaw** 1 year, 4 months ago

**Selected Answer: B**

Seems B to me

upvoted 1 times

 **shahin\_am2** 1 year, 4 months ago

B. False.

According to the Terraform documentation, the `terraform_remote_state` data source can be used to access state data stored in a remote backend. This allows you to use output values from one Terraform configuration as input values for another configuration.

However, the local backend is not considered a remote backend, because it stores state data locally on disk. Therefore, you cannot use the `terraform_remote_state` data source to access state stored with the local backend.

upvoted 2 times

 **princajen** 1 year, 4 months ago

**Selected Answer: B**

B. False.

The `terraform_remote_state` data source is used to fetch outputs from the state of a different Terraform configuration, which is stored remotely and used as input for the current configuration. It is used with remote backends, not local ones.

upvoted 2 times

✉️  **pyro7** 1 year, 5 months ago

False. The local backend stores state locally on the same machine as the Terraform configuration files, and it is not accessible over a network connection. To access state stored in a remote backend, you need to use the `terraform_remote_state` data source.

upvoted 1 times

✉️  **Abuu** 1 year, 5 months ago

**Selected Answer: A**

this is correct. The `terraform_remote_state` data source allows you to access state stored with the local backend, enabling you to share and reference state between different configurations. This is especially useful when using multiple configurations to manage different parts of the same infrastructure.

upvoted 2 times

✉️  **Daro\_** 1 year, 5 months ago

**Selected Answer: A**

As per below documentation

<https://developer.hashicorp.com/terraform/language/state/remote-state-data>

and example of configuration

```
data "terraform_remote_state" "vpc" {
 backend = "local"
```

```
 config = {
 path = "..."
 }
}
```

Answer is: [ A ]

upvoted 4 times

✉️  **lezgino** 1 year, 5 months ago

The `terraform_remote_state` data source is used to access state stored with a remote backend, such as S3 or Consul. You cannot access state stored with the local backend using the `terraform_remote_state` data source.

upvoted 1 times

✉️  **Daro\_** 1 year, 5 months ago

As per below documentation

<https://developer.hashicorp.com/terraform/language/state/remote-state-data>

and example of configuration

```
data "terraform_remote_state" "vpc" {
 backend = "local"
```

```
 config = {
 path = "..."
 }
}
```

Answer is: [ A ]

upvoted 2 times

✉️  **kounilasco** 1 year, 5 months ago

**Selected Answer: B**

it is false

upvoted 2 times

✉️  **gekkeheng** 1 year, 6 months ago

**Selected Answer: B**

`terraform state` is the right command. That can be used with "list" or "show".

upvoted 2 times

Question #195

*Topic 1*

You have been working in a Cloud provider account that is shared with other team members. You previously used Terraform to create a load balancer that is listening on port 80. After some application changes, you updated the Terraform code to change the port to 443.

You run `terraform plan` and see that the execution plan shows the port changing from 80 to 443 like you intended, and step away to grab some coffee.

In the meantime, another team member manually changes the load balancer port to 443 through the Cloud provider console before you get back to your desk.

What will happen when you `terraform apply` upon returning to your desk?

- A. Terraform will fail with an error because the state file is no longer accurate.
- B. Terraform will change the load balancer port to 80, and then change it back to 443.
- C. Terraform will not make any changes to the Load Balancer and will update the state file to reflect any changes made.
- D. Terraform will change the port back to 80 in your code.

**Correct Answer: C**

### Community vote distribution

C (100%)

✉️ **shanker\_sumit** 9 months, 2 weeks ago  
option A.

Terraform will fail with an error because the state file is no longer accurate.  
As person has made the changes via the console not through terraform .

Hence , it give an error as changes are made outside of terraform.  
upvoted 1 times

✉️ **camps** 1 year, 3 months ago

**Selected Answer: C**

It's C

upvoted 1 times

✉️ **sribalaje** 1 year, 5 months ago

**Selected Answer: C**

C is correct answer

upvoted 1 times

✉️ **gekkeheng** 1 year, 6 months ago

**Selected Answer: C**

As the state is refreshed during the "apply" no changes will be made on the cloud. Terraform will rather update its state file.

upvoted 4 times

✉️ **Daro\_** 1 year, 5 months ago

terraform apply command is not refreshing state, to detect drift without changing state file you need to type:  
terraform plan -refresh-only

terraform apply, will change state file according to configuration file  
terraform  
upvoted 1 times

✉️ **Misiek** 10 months, 1 week ago

The terraform apply command executes the actions proposed in a Terraform plan.  
Since the plan hasn't been saved the plan will run again with apply.

Command: plan

The terraform plan command creates an execution plan, which lets you preview the changes that Terraform plans to make to your infrastructure. By default, when Terraform creates a plan it:

Reads the current state of any already-existing remote objects to make sure that the Terraform state is up-to-date.  
Compares the current configuration to the prior state and noting any differences.  
Proposes a set of change actions that should, if applied, make the remote objects match the configuration.  
upvoted 1 times

✉️ **David\_C\_90** 1 year, 4 months ago

When you run terraform apply without passing a saved plan file, Terraform automatically creates a new execution plan as if you had run terraform plan, prompts you to approve that plan, and takes the indicated actions.

<https://developer.hashicorp.com/terraform/cli/commands/apply#automatic-plan-mode>  
upvoted 2 times

In a Terraform Cloud workspace linked to a version control repository, speculative plan runs start automatically when you merge or commit changes to version control.

- A. True
- B. False

**Correct Answer: A***Community vote distribution*

A (71%)

B (29%)

**vortegon** 7 months, 2 weeks ago**Selected Answer: B**

pull requests start speculative plans, this occurs before a commit or merge.  
<https://developer.hashicorp.com/terraform/cloud-docs/run/remote-operations#speculative-plans>  
upvoted 2 times

**FarziWaliMarzi** 1 year, 2 months ago**Selected Answer: A**

<https://developer.hashicorp.com/terraform/cloud-docs/run/ui#automatically-starting-runs>  
upvoted 2 times

**princajen** 1 year, 4 months ago**Selected Answer: A**

It is true because Terraform Cloud can monitor the linked version control repository for changes and automatically trigger a speculative plan run in response to each commit or merge. This allows users to quickly see the expected changes that would result from the proposed change, without actually applying those changes. Speculative plan runs can be a useful tool for catching errors early in the development process and avoiding potentially costly mistakes.

upvoted 2 times

**InformationOverload** 1 year, 6 months ago**Selected Answer: A**

Whether to perform speculative plans on pull requests to the connected repository, to assist in reviewing proposed changes. Automatic speculative plans are enabled by default, but you can disable them for any workspace.

<https://developer.hashicorp.com/terraform/cloud-docs/workspaces/settings/vcs>  
upvoted 1 times

You have some Terraform code and a variable definitions file named dev.auto.tfvars that you tested successfully in the dev environment. You want to deploy the same code in the staging environment with a separate variable definition file and a separate state file.

Which two actions should you perform? (Choose two.)

- A. Copy the existing terraform.tfstate file and save it as staging.terraform.tfstate
- B. Write a new staging.auto.tfvars variable definition file and run Terraform with the var-file="staging.auto.tfvars" flag
- C. Create a new Terraform workspace for staging
- D. Create a new Terraform provider for staging
- E. Add new Terraform code (\*.tf files) for staging in the same directory

**Correct Answer: BC***Community vote distribution*

BC (100%)

**7b5b962** 4 weeks ago

B. Terraform core

Terraform core is responsible for reading the configuration, generating the execution plan, and applying the changes by interacting with the providers.

upvoted 1 times

**starksolutions** 7 months, 2 weeks ago

merci .

upvoted 1 times

**akm\_1010** 11 months, 1 week ago**Selected Answer: BC**

with a separate variable definition file :- Write a new staging.auto.tfvars variable definition file and run Terraform with the var-file="staging.auto.tfvars" flag

Want to deploy the same code in the staging environment with a separate state file :- Create a new Terraform workspace for staging. Worksp isolate Terraform state.

upvoted 2 times

**Ni33** 1 year, 2 months ago**Selected Answer: BC**

B and C are correct answers. same tfstate file can be used in multiple workspaces to provision infrastructure and each workspace has its own tfvar file matching workspace prefix.

upvoted 4 times

**camps** 1 year, 3 months ago**Selected Answer: BC**B. Write a new staging.auto.tfvars variable definition file and run Terraform with the var-file="staging.auto.tfvars" flag  
C. Create a new Terraform workspace for staging

upvoted 1 times

**ozbeyucel** 1 year, 5 months ago**Selected Answer: BC**

.....

upvoted 2 times

**nakikoo** 1 year, 6 months ago

Correct, workspace to use same .tf configuration with a different environment such as dev, prod, test,

write new stage then refer the existing .tf config

upvoted 1 times

The \_\_\_\_\_ determines how Terraform creates, updates, or deletes resources.

- A. Terraform configuration
- B. Terraform core
- C. Terraform provider
- D. Terraform provisioner

**Correct Answer: A**

*Community vote distribution*

C (79%)

A (21%)

 **gekkehenk** Highly Voted 1 year, 6 months ago

**Selected Answer: C**

The question specifically states "how". The provider is the only component of Terraform that know HOW to create, update, delete a resource, it knows all the specifics.

upvoted 18 times

 **Manguu** Highly Voted 1 year, 3 months ago

**Selected Answer: C**

"What" = config  
"How" = provider  
upvoted 13 times

 **7b5b962** Most Recent 4 weeks ago

B. Terraform core

Terraform core is responsible for reading the configuration, generating the execution plan, and applying the changes by interacting with the providers.

upvoted 1 times

 **Stanislav4907** 1 year, 3 months ago

**Selected Answer: C**

The Terraform provider determines how Terraform creates, updates, or deletes resources.

Providers are responsible for translating Terraform configurations into API requests that manipulate resources in the target environment. Each provider implements the necessary operations for a specific type of infrastructure, such as AWS, Azure, or Google Cloud Platform.

When Terraform applies a configuration, it reads the provider information from the configuration and uses it to interact with the target environment. The provider then manages the lifecycle of the resources by creating, updating, or deleting them as needed.

Note that while the Terraform configuration defines the desired state of the resources, and the Terraform core is responsible for managing the planning and execution of changes, it is ultimately the provider that determines how those changes are implemented in the target environment.

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: C**

C. Terraform provider

upvoted 2 times

 **Atila50** 1 year, 4 months ago

**Selected Answer: A**

<https://developer.hashicorp.com/terraform/language/resources/behavior>

upvoted 2 times

 **David\_C\_90** 1 year, 4 months ago

Terraform configuration determines the "what" i.e. what infrastructure to build. Not the "how"; that's the whole point behind IaC

upvoted 3 times

 **rotimislaw** 1 year, 4 months ago

**Selected Answer: C**

as gekkehenk wrote, it's C

upvoted 2 times

 **princajen** 1 year, 4 months ago

**Selected Answer: C**

The Terraform provider determines how Terraform interacts with a specific API or service provider to create, update, or delete resources. The provider translates Terraform configuration files into API requests, and then interacts with the API to manage resources on the service provider

upvoted 1 times

 **crickmeister** 1 year, 5 months ago

C. Terraform provider determines how Terraform creates, updates, or deletes resources.

A provider is responsible for understanding API interactions and exposing resources. Providers can represent physical resources like compute instances, or abstract resources like DNS records. A provider is responsible for translating the Terraform configuration into API calls to the underlying service, and for handling the response from the service.

The Terraform configuration specifies which provider to use for each resource block. The provider block itself specifies the configuration necessary to connect to the API of a particular service, such as credentials or endpoint information.

upvoted 1 times

 **agmesas** 1 year, 5 months ago

**Selected Answer: C**

C, provider = how, .tf = what

upvoted 1 times

 **Abuu** 1 year, 5 months ago

**Selected Answer: A**

The Terraform configuration is the main input for Terraform, defining the desired state of the resources in the infrastructure. It is the definition of how Terraform should create, update, or delete resources in order to reach the desired state.

upvoted 1 times

 **dkd123** 1 year, 5 months ago

Terraform config determines what , Provider determines how. Answer should be provider.

upvoted 3 times

 **ozbeyucel** 1 year, 5 months ago

**Selected Answer: A**

correct is A

upvoted 2 times

 **Abuu** 1 year, 5 months ago

**Selected Answer: A**

State file should be included; because The State File determines how Terraform creates, updates, or deletes resources. The state file is a JSON file that is used to persistently store all the resources created by Terraform. It stores the state of the resources, as well as their properties, so Terraform knows what actions to take on the next run.

upvoted 1 times

 **Abuu** 1 year, 5 months ago

Applying a Terraform configuration is the process of creating, updating, and destroying real infrastructure objects in order to make their settings match the configuration.

<https://developer.hashicorp.com/terraform/language/resources/behavior>

upvoted 1 times

 **Agil09** 1 year, 5 months ago

**Selected Answer: A**

correct is A

upvoted 2 times

 **Zeppoonstream** 1 year, 5 months ago

**Selected Answer: A**

A. Terraform configuration

The Terraform configuration determines how Terraform creates, updates, or deletes resources. It is written in HashiCorp Configuration Language (HCL) and it includes the resources that will be managed by Terraform, their properties, and the dependencies between them. The configuration is read by the Terraform core and used to generate an execution plan that is then passed to the Terraform provider to make the necessary changes to the infrastructure.

upvoted 1 times

Terraform destroy is the only way to remove infrastructure.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **ramzez4815** 8 months, 3 weeks ago

**Selected Answer: B**

False: the primary way to delete infrastructure managed by Terraform is to remove the corresponding declarations from your configuration and run terraform apply.

terraform destroy (or, equivalently, terraform apply -destroy) is for the less common situation where you want to destroy everything managed by a particular configuration, which is not an everyday operation for long-lived infrastructure but can be useful for short-lived infrastructure such as one-time testing environments.

<https://discuss.hashicorp.com/t/is-terraform-destroy-the-only-way/36619/4>

upvoted 1 times

 **Basavaraju\_V** 1 year, 4 months ago

B. by commenting the resource block in the configuration file will also destroy the resources

upvoted 3 times

 **Pietjeplukgeluk** 1 year, 5 months ago

"terraform apply -destroy" would be an alias for "terraform destroy" B for sure

upvoted 2 times

 **ArizonaClassics** 1 year, 6 months ago

B for Sure

upvoted 1 times

 **nakikoo** 1 year, 6 months ago

correct, you can also remove the resource definition in the state file if you previously stated it, it will detect the resource definition is absent and will delete the resource

Unless you state the resource, but someone delete it manually, it will recreate it

upvoted 1 times

 **JohnGasp** 11 months, 3 weeks ago

if you remove just from state, terraform will forgot about resource and just create new one on next apply, so first instance will not be deleted

upvoted 1 times

Which of the following is the correct way to pass the value in the variable num\_servers into a module with the input servers in HCL2?

- A. servers - var.num\_servers
- B. servers - num\_servers
- C. servers - var(num\_servers)
- D. \$(var.num\_servers)

**Correct Answer: B**

*Community vote distribution*

A (100%)

 **LeyLey** Highly Voted 1 year, 5 months ago

Answer is A but is should be servers = var.num\_servers  
upvoted 6 times

 **Nunyabiznes** Most Recent 1 year, 3 months ago

Selected Answer: A  
Selected Answer: A  
module "example" {  
source = "path/to/module"  
servers = var.num\_servers  
}  
upvoted 4 times

 **Basavaraju\_V** 1 year, 4 months ago

by commenting the resource block in configuration file will also destroy the resources  
upvoted 1 times

 **princajen** 1 year, 4 months ago

Selected Answer: A  
Selected Answer: A  
The correct way to pass the value in the variable num\_servers into a module with the input servers in HCL2 is as follows:  
module "example" {  
source = "./modules/example"  
servers = var.num\_servers  
}  
upvoted 1 times

 **Daro\_** 1 year, 5 months ago

Selected Answer: A  
Selected Answer: A  
servers = var.num\_servers  
upvoted 2 times

 **Zeppoonstream** 1 year, 5 months ago

Selected Answer: A  
Selected Answer: A  
A. servers - var.num\_servers

In HCL2, the correct way to pass the value in the variable num\_servers into a module with the input servers is by using the following syntax:  
servers = var.num\_servers

This tells Terraform to use the value of the variable num\_servers as the value of the input servers in the module.  
upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

correct way to pass the value of the variable num\_servers into a module with the input servers is to use the var syntax in the module block:

```
module "module_example" {
 source = "path/to/module"
 servers = var.num_servers
}
```

upvoted 2 times

 **resnef** 1 year, 6 months ago

going with A : servers = var.num\_servers

upvoted 1 times

 **gekkeheng** 1 year, 6 months ago

**Selected Answer: A**

Variables are always referenced with a var. prefix.

upvoted 1 times

 **Edileimig** 1 year, 6 months ago

should be A ?

upvoted 1 times

 **nakikoo** 1 year, 6 months ago

not sure if A or B, hope someone may clarify this

upvoted 2 times

Question #201

*Topic 1*

Which of the following commands would you use to access all of the attributes and details of a resource managed by Terraform?

- A. terraform state list
- B. terraform state show
- C. terraform get
- D. terraform state list

**Correct Answer: A**

*Community vote distribution*

B (100%)

 **Albion** 7 months, 1 week ago

**Selected Answer: B**

terraform state list : is used to list all resources that are currently being tracked in the Terraform state. It provides a list of resource addresses, which can be helpful to identify the names or identifiers of resources managed by Terraform.

terraform state show : command is used to show details and attributes of a resource managed by Terraform. It provides a human-readable representation of the current state of a specific resource, including all its attributes and configuration.

Correct answer : B

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

B. terraform state show

upvoted 1 times

 **sagunala5** 1 year, 3 months ago

B

<https://developer.hashicorp.com/terraform/cli/commands/state/show>

upvoted 1 times

 **mhinojosarubia** 1 year, 5 months ago

**Selected Answer: B**

It's b

upvoted 1 times

 **Abuu** 1 year, 5 months ago

B defininitely

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: B**

The terraform state show command is used to show the attributes of a single resource in the Terraform state.

<https://developer.hashicorp.com/terraform/cli/commands/state/show>

upvoted 2 times

 **gekkeheng** 1 year, 6 months ago

**Selected Answer: B**

As they're are asking about "details" it should be B.

upvoted 1 times

 **srcntp** 1 year, 6 months ago

**Selected Answer: B**

The terraform state list command shows the resource addresses for every resource Terraform knows about in a configuration, optionally filtered by partial resource address. The terraform state show command displays detailed state data about one resource.

upvoted 2 times

 **ssanjayt** 1 year, 6 months ago

**Selected Answer: B**

B is correct

upvoted 1 times

 **nakikoo** 1 year, 6 months ago

first!, joking we're intellectuals, its A

upvoted 2 times

How would you be able to reference an attribute from the vsphere\_datacenter data source for use with the datacenter\_id argument within the vsphere\_folder resource in the following configuration?

```
data "vsphere_datacenter" "dc" {}

resource "vsphere_folder" "parent" {
 path = "Production"
 type = "vm"
 datacenter id = _____
}
```

- A. data.dc.id
- B. data.vsphere\_datacenter.dc
- C. vsphere\_datacenter.dc.id
- D. data.vsphere\_datacenter.dc.id

**Correct Answer: D**

*Community vote distribution*

D (100%)

✉  **resnef** Highly Voted 1 year, 6 months ago

data.<TYPE>.<NAME>.<ATTRIBUTE>  
upvoted 8 times

✉  **mamoon\_malta2022** Most Recent 1 year, 4 months ago

D:  
data "azurerm\_resource\_group" "example" {  
name = "existing"  
}  
  
output "id" {  
value = data.azurerm\_resource\_group.example.id  
}  
upvoted 3 times

✉  **InformationOverload** 1 year, 6 months ago

**Selected Answer: D**  
D is correct  
upvoted 2 times

✉  **ArizonaClassics** 1 year, 6 months ago

I will go for D  
upvoted 2 times

✉  **nakikoo** 1 year, 6 months ago

yea correct, D,  
  
provider.resourcetype.name.id  
upvoted 2 times

You decide to move a Terraform state file to Amazon S3 from another location. You write the code below into a file called backend.tf.

```
terraform {
 backend "s3" {
 bucket = "my-tf-bucket"
 region = "us-east-1"
 }
}
```

Which command will migrate your current state file to the new S3 remote backend?

- A. terraform state
- B. terraform init
- C. terraform refresh
- D. terraform push

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Darshan07** 4 months, 2 weeks ago

**Selected Answer: B**

I will go with B

upvoted 1 times

 **gekkehenk** 1 year, 6 months ago

**Selected Answer: B**

Answer is correct, terraform init migrates the state.

upvoted 2 times

 **ArizonaClassics** 1 year, 6 months ago

I will go for B

upvoted 1 times

 **nakikoo** 1 year, 6 months ago

- A. terraform state just state your .tf
- B. Terraform init synch with backend
- C. Refresh, refreshes config
- D. Push, not sure

upvoted 1 times

You want to tag multiple resources with a string that is a combination of a generated random\_id and a variable.

How should you use the same value in all these resources without repeating the random\_id and variable in each resource?

- A. Local values
- B. Data source
- C. Modules
- D. Outputs

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **ArizonaClassics** Highly Voted 1 year, 6 months ago

A is correct: Local values: A local value assigns a name to an expression, so you can use the name multiple times within a module instead of repeating the expression.

<https://developer.hashicorp.com/terraform/language/values/locals>

upvoted 5 times

 **Darshan07** Most Recent 4 months, 2 weeks ago

Selected Answer: A

A is the correct option

upvoted 1 times

 **camps** 1 year, 3 months ago

Selected Answer: A

A. Local values

upvoted 3 times

 **lezgino** 1 year, 5 months ago

A is correct

upvoted 2 times

Which of the following is not a benefit of adopting infrastructure as code?

- A. Interpolation
- B. Reusability of code
- C. Versioning
- D. Automation

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Nani0107** 1 year ago

**Selected Answer: A**

A is correct

upvoted 1 times

 **Abuu** 1 year, 5 months ago

A is definitely correct

upvoted 2 times

 **ArizonaClassics** 1 year, 6 months ago

A is correct

upvoted 2 times

Module version is required to reference a module on the Terraform Module Registry.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (75%)

A (25%)

 **gekkeheneK**  1 year, 6 months ago

**Selected Answer: B**

Specifying a version is not mandatory. When a version is not specified, Terraform just downloads the latest version  
upvoted 10 times

 **Pietjeplukgeluk** 1 year, 5 months ago

Correct, you can run "terraform init" without stating the version. It is still recommended to specify the version, but it is not mandatory.  
upvoted 3 times

 **master9** Most Recent 4 days, 11 hours ago

**Selected Answer: A**

When referencing a module from the Terraform Module Registry, specifying a version is required. This ensures that your infrastructure code is consistent and reproducible by always using the same version of the module. Here is an example of how you would reference a module with a specific version:

```
module "vpc" {
 source = "terraform-aws-modules/vpc/aws"
 version = "2.77.0"
```

```
... other module variables ...
}
```

upvoted 1 times

 **yubac** 1 year, 1 month ago

**Selected Answer: B**

it is not mandatory. If not set terraform will download the latest (not recommended)

upvoted 1 times

 **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: B**

Module version is not required to reference a module on the Terraform Module Registry. If you don't specify a version, Terraform will use the latest version available. However, it is a good practice to specify a version to ensure that your configuration continues to work with the same version of the module, even if newer versions are released.

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

It's B - False

upvoted 1 times

 **Atila50** 1 year, 4 months ago

**Selected Answer: A**

when fetching a model version is required

upvoted 1 times

 **Errorinprocess** 1 year, 4 months ago

**Selected Answer: B**

Selecting B:

<https://developer.hashicorp.com/terraform/tutorials/modules/module-use>

The version argument is not required, but we highly recommend you include it when using a Terraform module. For supported sources, this argument specifies the module version Terraform will load. Without the version argument, Terraform will load the latest version of the module.

upvoted 1 times

 **jabli** 1 year, 4 months ago

**Selected Answer: B**

By default, Terraform will clone and use the default branch (referenced by HEAD) in the selected repository. You can override this using the ref argument.

upvoted 1 times

 **princajen** 1 year, 4 months ago

Yes, it is required. According to the official Terraform documentation:

"When you use a module from the Terraform Module Registry, you'll need to specify both the source address and the module version, which will be in the format {NAMESPACE}/{MODULE NAME}/{PROVIDER}/{VERSION}." (source: <https://www.terraform.io/docs/language/modules/sources.html>)

upvoted 1 times

 **princajen** 1 year, 4 months ago

**Selected Answer: A**

This is true because Terraform Modules are versioned, and versioning ensures that the module being used is stable and won't change unexpectedly. When referencing a module on the Terraform Module Registry, you must specify the module source address and the version of the module to use. This is usually done in the module block of your Terraform configuration.

upvoted 1 times

✉️  **pyro7** 1 year, 5 months ago

True. When referencing a module from the Terraform Module Registry, you must specify a version constraint to indicate which version of the module you want to use. The version constraint is specified in the version argument in the module block. This allows you to lock in a specific version of a module to ensure stability and consistency in your Terraform configurations.

upvoted 1 times

✉️  **nhvardhan** 1 year, 5 months ago

**Selected Answer: A**

Module Versions are required to reference a correct module.

upvoted 2 times

✉️  **Only5** 1 year, 5 months ago

I think answer should be True, when you want to refer a particular module version is mandatory to use . below answer is from chat GPT Yes, when referencing a module on the Terraform Module Registry, the module version is required. This is because modules on the registry are versioned, and different versions of a module may have different configurations and behaviors.

upvoted 1 times

✉️  **Abuu** 1 year, 5 months ago

Module versions are not required but are recommended to avoid unexpected or unwanted changes.

Question #207

Topic 1

While deploying a virtual machine, the first launch user\_data script fails due to race condition with another resource deployed during the same Terraform run.

What is the least disruptive method to correct the issue?

- A. Run terraform taint against the virtual machine's resource name, then terraform apply
- B. Restart the virtual machine from the cloud portal
- C. Run terraform apply again
- D. Run terraform destroy then terraform apply

**Correct Answer: A**

*Community vote distribution*

A (53%)

C (47%)

✉️  **zaaath**  1 year, 3 months ago

I'd go with A.

If a user\_data script fails during the creation of an EC2 instance, Terraform will not mark the resource as tainted automatically. This is because Terraform does not have visibility into the success or failure of the user\_data script, as it is executed within the instance itself, and not directly controlled by Terraform.

However, if the user\_data script failure causes the instance to become unresponsive or otherwise unusable, Terraform may detect this during subsequent plan or apply operation and mark the instance as "tainted". A tainted resource is one that Terraform cannot verify the current state and must be recreated in order to bring it back to a known state.

upvoted 6 times

✉️  **Mantis**  1 year, 2 months ago

**Selected Answer: A**

The script deployment will not have been tracked within Terraform - and will therefore not re-run on the next 'apply' command. The resource t script is running on will have to be redeployed to trigger the script to run a second time - the least disruptive way do achieve this is with the 't command

upvoted 5 times

✉️  **alex78** Most Recent 5 months, 2 weeks ago

**Selected Answer: C**

If a creation-time provisioner fails, the resource is marked as tainted. A tainted resource will be planned for destruction and recreation upon the next terraform apply.

<https://developer.hashicorp.com/terraform/language/resources/provisioners/syntax>

upvoted 2 times

✉️  **March2023** 1 year ago

**Selected Answer: A**

going with A

upvoted 1 times

✉️  **Ni33** 1 year, 2 months ago

A is correct

upvoted 2 times

✉️  **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: C**

C. Run terraform apply again

upvoted 2 times

✉️  **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: C**

if it fails, you just re-run terraform apply again , why even run taint for something that didn't even get created in the first place.

For example, you're trying to create Resource A, but it fails because resource B was being created. So what do you do to create resource A a ? - terraform apply

upvoted 3 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer: A**

In this scenario, the user\_data script failed due to a race condition with another resource being deployed simultaneously during the same Terraform run. The least disruptive method to correct the issue is to use the terraform taint command followed by terraform apply.

Running terraform taint against the virtual machine's resource name marks the resource as tainted. Tainting a resource indicates that it has a problem and should be destroyed and recreated during the next terraform apply. By doing this, you force Terraform to recreate the virtual machine with the corrected user\_data script, resolving the race condition without affecting other resources.

upvoted 2 times

✉️  **Pietjeplukgeluk** 1 year, 5 months ago

The question leaves room for interpretation. If a creation-time provisioner fails, the resource is marked as tainted. A tainted resource will be planned for destruction and recreation upon the next terraform apply.

<https://developer.hashicorp.com/terraform/language/resources/provisioners/syntax>

upvoted 1 times

✉️  **David\_C\_90** 1 year, 4 months ago

The question states user\_data not provisioners...

upvoted 2 times

✉️  **Abuu** 1 year, 5 months ago

A is the Answer since the particular object has become degraded or damaged and Terraform taint will propose to replace it in the next plan you create.

upvoted 2 times

✉️  **ArizonaClassics** 1 year, 6 months ago

I will go for A

upvoted 2 times

The public Module Registry is free to use.

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **David\_C\_90** 1 year, 4 months ago

**Selected Answer: A**

Both public and private registries are free.

"Terraform Cloud includes a private registry that is available to all accounts, including free organizations."

<https://developer.hashicorp.com/terraform/registry/private#terraform-cloud-private-registry>

"Anyone can publish and consume providers, modules, and policies on the public Terraform Registry"

<https://developer.hashicorp.com/terraform/registry>

upvoted 2 times

 **Abuu** 1 year, 5 months ago

**Selected Answer: A**

<https://developer.hashicorp.com/terraform/registry/private#terraform-cloud-private-registry>

upvoted 1 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

<https://registry.terraform.io/browse/modules>

upvoted 1 times

 **ArizonaClassics** 1 year, 6 months ago

A is Correct. please disregard B

upvoted 3 times

 **Edileimig** 1 year, 6 months ago

yes, B

upvoted 1 times

 **ArizonaClassics** 1 year, 6 months ago

B is correct

upvoted 1 times

Both Terraform Cloud and Terraform Enterprise support policy as code (Sentinel).

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

A (100%)

✉️ **SHERLOCKAWS** 2 months, 3 weeks ago

**Selected Answer: A**

Answer is A. Since v0.25 supported on both Cloud and Enterprise >> <https://developer.hashicorp.com/sentinel/docs/terraform>  
upvoted 1 times

✉️ **Alex1atd** 9 months, 1 week ago

**Selected Answer: A**

Actually, is for both. Limited of course for Cloud free, but is available:  
<https://www.hashicorp.com/products/terraform/pricing>  
upvoted 2 times

✉️ **David\_C\_90** 1 year, 4 months ago

Sentinel is only available on Cloud Team & Governance, Cloud Business, and Enterprise. It is not available for Cloud Free.

Is this an old question?

upvoted 4 times

✉️ **ArizonaClassics** 1 year, 6 months ago

A is correct

upvoted 1 times

You want to define multiple data disks as nested blocks inside the resource block for a virtual machine.

What Terraform feature would help you define the blocks using the values in a variable?

- A. Local values
- B. Collection functions
- C. Dynamic blocks
- D. Count arguments

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **camps** 1 year, 3 months ago

**Selected Answer: C**

- C. Dynamic blocks  
upvoted 1 times

 **Abuu** 1 year, 5 months ago

**Selected Answer: C**

The correct answer is C. Dynamic Blocks. Dynamic blocks allow you to define multiple data disks as nested blocks inside the resource block a virtual machine by using the values in a variable. Dynamic blocks are used to create multiple resource instances from a single configuration block by iterating over the values of a list or map.

upvoted 2 times

 **ArizonaClassics** 1 year, 6 months ago

- C is correct  
upvoted 3 times

Which of the following module source paths does not specify a remote module?

- A. source = "./modules/consul"
- B. source = "git@github.com:hashicorp/example.git"
- C. source = "github.com/hashicorp/example"
- D. source = "hashicorp/consul/aws"

**Correct Answer: C**

*Community vote distribution*

A (100%)

 **SilentH** 2 months, 3 weeks ago

**Selected Answer: A**

Honestly, is Examtopics trying to troll us with their answers?

upvoted 1 times

 **Agil09** 1 year, 5 months ago

**Selected Answer: A**

A correct

upvoted 2 times

 **InformationOverload** 1 year, 6 months ago

**Selected Answer: A**

A - since its local path

upvoted 3 times

 **Edileimig** 1 year, 6 months ago

**Selected Answer: A**

A is right answer

upvoted 1 times

 **resnef** 1 year, 6 months ago

**Selected Answer: A**

question says "remote" , hence A

upvoted 1 times

 **resnef** 1 year, 6 months ago

not remote = local

upvoted 1 times

 **gekkehenk** 1 year, 6 months ago

**Selected Answer: A**

The specification of a path, indentifies a local module. The others specify a remote path. Therefore A is the right answer.

upvoted 2 times

 **ArizonaClassics** 1 year, 6 months ago

I would go for A

see <https://developer.hashicorp.com/terraform/language/modules/sources>

upvoted 1 times

You have a list of numbers that represents the number of free CPU cores on each virtual cluster:

```
numcpus = [18, 3, 7, 11, 2]
```

What Terraform function could you use to select the largest number from the list?

- A. max(numcpus)
- B. ceil(numcpus)
- C. top(numcpus)
- D. high[numcpus]

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **gekkehenk** 1 year, 6 months ago

**Selected Answer: A**

"max takes one or more numbers and returns the greatest number from the set."

upvoted 3 times

 **ArizonaClassics** 1 year, 6 months ago

A is correct

See <https://developer.hashicorp.com/terraform/language/functions/max>

upvoted 1 times

Variables declared within a module are accessible outside of the module.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (60%)

A (40%)

✉️ **Mail1964** Highly Voted 12 months ago

**Selected Answer: A**

It is a poorly worded Q. Using output they are accessible and not without. But the Q only asks if they "are accessible" which is A. If the Q add "by default" on the end, then it would be B.

upvoted 8 times

✉️ **camps** Highly Voted 1 year, 3 months ago

**Selected Answer: B**

B. False

Variables declared within a module are not accessible outside of the module by default. To expose the values of those variables to other modules or the root configuration, you need to use output values. Output values are declared using the output block in the module, and they serve as a way to expose certain values from a module so that they can be consumed by other modules or configurations.

upvoted 5 times

✉️ **princajen** Most Recent 1 year, 4 months ago

**Selected Answer: B**

B. False.

Variables declared within a module have module-level scope and can only be accessed within the module. If a variable in a module needs to be accessed outside of the module, it must be exposed as an output.

upvoted 3 times

✉️ **lezgino** 1 year, 5 months ago

B is correct

upvoted 1 times

✉️ **mocnak** 1 year, 5 months ago

It is a bit confusing question, as you pointed out, they are accessible if they are passed out as output variables. so yes, they are. I am not sure that's the "catch" of this question ..

upvoted 4 times

✉️ **InformationOverload** 1 year, 6 months ago

**Selected Answer: B**

these variables are not automatically accessible outside of the module unless they are explicitly passed out as output variables

upvoted 4 times

Which of the following is not a valid Terraform variable type?

A. list

- B. map
- C. array
- D. string

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **InformationOverload** Highly Voted 1 year, 6 months ago

**Selected Answer: C**

Array is not a variable type. Variable types in terraform are:

string: A sequence of Unicode characters

number: A numeric value, either an integer or a floating-point number

bool: A Boolean value of either true or false

list: An ordered collection of values, all of which must be of the same type

map: A collection of key-value pairs, where the keys and values can be of any type

set: An unordered collection of unique values, all of which must be of the same type

object: A complex structure that can contain other objects, lists, and scalar values

tuple: A fixed-length array of specific types. (Terraform version 0.13 and later)

<https://developer.hashicorp.com/terraform/language/expressions/types>

upvoted 5 times

 **Derag** Most Recent 9 months, 3 weeks ago

C, array is not a variable type.

upvoted 1 times

 **Jlee7** 1 year ago

C is the correct answer.

upvoted 1 times

 **agmesas** 1 year, 5 months ago

**Selected Answer: C**

array is not

upvoted 1 times

What is a key benefit of the Terraform state file?

- A. A state file can be used to schedule recurring infrastructure tasks
- B. A state file represents a source of truth for resources provisioned with a public cloud console
- C. A state file represents the desired state expressed by the Terraform code files
- D. A state file represents a source of truth for resources provisioned with Terraform

**Correct Answer: A**

*Community vote distribution*

D (100%)

 Blitz123 5 months, 2 weeks ago

Though the Answer is "D", he is the tricky aspect

Lets just say, we provisioned Infrastructure through Terraform using config files, then someone goes to CLI and makes a manual change. Once you apply terraform Refresh, you no longer have all the information in statefile which is provisioned through terraform.

Just like option C, you make changes to config files and you apply, desired state, current state are same and state file does represent the desired state expressed by the Terraform code files.

upvoted 1 times

 camps 1 year, 3 months ago

**Selected Answer: D**

it's D

upvoted 1 times

 Agil09 1 year, 5 months ago

**Selected Answer: D**

D is correct

upvoted 3 times

 Zeppoonstream 1 year, 5 months ago

**Selected Answer: D**

D. A state file represents a source of truth for resources provisioned with Terraform

A key benefit of the Terraform state file is that it represents a source of truth for resources provisioned with Terraform. The state file is used to keep track of the current state of the infrastructure resources that are being managed by Terraform. It contains information about the resources, their properties, and the dependencies between them.

The state file is used by Terraform to determine what changes need to be made to the infrastructure to reach the desired state defined in the configuration files.

- A. A state file can be used to schedule recurring infrastructure tasks, is not a benefit of state file.
- B. A state file represents a source of truth for resources provisioned with a public cloud console, is not a benefit of state file.
- C. A state file represents the desired state expressed by the Terraform code files, is not a benefit of state file.

upvoted 1 times

 Nirms 1 year, 6 months ago

**Selected Answer: D**

D is correct

upvoted 1 times

 gekkehenk 1 year, 6 months ago

**Selected Answer: D**

The idea of a state file is to operate as a source of truth. If the infrastructure deviates from that it is capable of fixing that.

upvoted 2 times

Question #216

Topic 1

Which of these statements about Terraform Enterprise workspaces is false?

- A. They can securely store cloud credentials
- B. You must use the CLI to switch between workspaces
- C. Plans and applies can be triggered via version control system integrations
- D. They have role-based access controls

**Correct Answer: C**

*Community vote distribution*

B (100%)

 **lezgino** Highly Voted 1 year, 5 months ago

B. You must use the CLI to switch between workspaces is false.

Terraform Enterprise provides a web-based UI that allows you to switch between workspaces, view the state of your infrastructure, and run Terraform commands without having to use the command line interface.

- A. They can securely store cloud credentials is true.
- C. Plans and applies can be triggered via version control system integrations is true.
- D. They have role-based access controls is true.

upvoted 8 times

 **wirkmood** Most Recent 2 months ago

**Selected Answer: B**

It's B for sure

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

B. You must use the CLI to switch between workspaces

This statement is false. In Terraform Enterprise, workspaces are designed to be managed through the web UI, API, or version control system (VCS) integrations, rather than using the Terraform CLI to switch between them. Each workspace in Terraform Enterprise corresponds to a specific environment or configuration, and you can manage them using the available integrations, allowing for better collaboration and automation.

upvoted 1 times

 **gekkeheneh** 1 year, 4 months ago

**Selected Answer: B**

Correct answer: B

The statement that is false is that "You must use the CLI to switch between workspaces" in Terraform Enterprise.

Terraform Enterprise provides a web-based UI that allows you to switch between workspaces, view the state of your infrastructure, and run Terraform commands without having to use the command line interface.

upvoted 2 times

 **rotimislaw** 1 year, 4 months ago

**Selected Answer: B**

As lezgino says, it's B

upvoted 1 times

 **Arrash** 1 year, 5 months ago

**Selected Answer: B**

B is correct

upvoted 1 times

 **pyro7** 1 year, 5 months ago

The correct answer is B

upvoted 1 times

 **PavelTech** 1 year, 5 months ago

**Selected Answer: B**

B you can switch workspaces from UI

upvoted 1 times

 **Zeppoonstream** 1 year, 5 months ago

**Selected Answer: B**

B. You must use the CLI to switch between workspaces

The statement that is false is that "You must use the CLI to switch between workspaces" in Terraform Enterprise.

Terraform Enterprise provides a web-based UI that allows you to switch between workspaces, view the state of your infrastructure, and run Terraform commands without having to use the command line interface.

- A. They can securely store cloud credentials is true.
- C. Plans and applies can be triggered via version control system integrations is true.
- D. They have role-based access controls is true.

upvoted 1 times

Define the purpose of state in Terraform.

- A. State is used to map real world resources to your configuration and keep track of metadata
- B. State is a method of codifying the dependencies of related resources
- C. State is used to enforce resource configurations that relate to compliance policies
- D. State is used to store variables and quickly reuse existing code

**Correct Answer:** B

*Community vote distribution*

A (100%)

 [Removed] 1 year ago

**Selected Answer: A**

Textbook Ans:

Terraform must store state about your managed infrastructure and configuration. This state is used by Terraform to map real world resources to your configuration, keep track of metadata, and to improve performance for large infrastructures.

<https://developer.hashicorp.com/terraform/language/state>

upvoted 2 times

 March2023 1 year ago

**Selected Answer: A**

metadata

upvoted 1 times

 FarziWaliMarzi 1 year, 2 months ago

**Selected Answer: A**

definitely A

upvoted 1 times

 bugalter 1 year, 3 months ago

**Selected Answer: A**

Selected Answer: A

upvoted 1 times

 Nunyabiznes 1 year, 3 months ago

**Selected Answer: A**

The purpose of state in Terraform is to map real-world resources to your configuration and keep track of metadata. Terraform state plays a crucial role in tracking the relationships between your configuration and the actual resources provisioned in your infrastructure. This allows Terraform to understand the current state of your infrastructure, manage changes over time, and perform operations such as updating, destroying, or importing resources.

upvoted 1 times

 camps 1 year, 3 months ago

**Selected Answer: A**

A. State is used to map real world resources to your configuration and keep track of metadata

In Terraform, state serves the purpose of mapping real-world resources to your configuration and maintaining metadata about those resources. The state file records the current state of the infrastructure that Terraform manages, allowing Terraform to detect any differences between the actual infrastructure and the desired state defined in the configuration files. This information is crucial for planning and applying changes to the infrastructure in a consistent and reliable manner.

upvoted 1 times

 tbhttp 1 year, 3 months ago

A. State is used to map real world resources to your configuration and keep track of metadata.

The purpose of state in Terraform is to keep track of the current state of your infrastructure as defined by your configuration files. When you apply your configuration, Terraform creates a plan that describes the changes needed to move from the current state to the desired state. Once you apply the plan, Terraform updates the state file to reflect the new state of your infrastructure. The state file maps the resources you have created in the real world to the configuration in your Terraform code, and it also stores metadata such as resource IDs and IP addresses that may be needed for future updates or maintenance. This allows Terraform to manage and modify your infrastructure in a consistent and reliable way, aiming to ensure that your resources are always in the desired state.

upvoted 1 times

 Ravi528 1 year, 3 months ago

A is not correct, State tries to match the resources managed by Terraform config files, but not the real world resources/infra. So B is correct

upvoted 1 times

 SilentMilli 1 year, 3 months ago

**Selected Answer: A**

The primary purpose of state is to map the desired state of your infrastructure as defined in your configuration to the actual state of your infrastructure in the real world. Terraform uses the state to determine what changes need to be made to bring the actual state of your infrastructure into line with the desired state specified in your configuration.

upvoted 1 times

Which backend does the Terraform CLI use by default?

- A. API
- B. Remote
- C. Terraform Cloud
- D. Local
- E. HTTP

**Correct Answer: D**

*Community vote distribution*

D (100%)

 awsexams 6 months, 4 weeks ago

**Selected Answer: D**

D. Local

upvoted 1 times

 Jhaggar 1 year, 2 months ago

**Selected Answer: D**

The Terraform CLI uses the local backend by default, which stores the state file on the local disk of the machine running Terraform.

upvoted 1 times

 FarziWaliMarzi 1 year, 2 months ago

**Selected Answer: D**

Local for sure

upvoted 1 times

 camps 1 year, 3 months ago

**Selected Answer: D**

D. Local

By default, the Terraform CLI uses the local backend to store the state file. The local backend stores the state as a file on your local filesystem you want to use a different backend, such as a remote backend (e.g., Terraform Cloud) or other options like S3 or Consul, you need to config the backend explicitly in your Terraform configuration files.

upvoted 1 times

 SilentMilli 1 year, 3 months ago

**Selected Answer: D**

The Terraform CLI uses a local backend by default, which stores the state of your infrastructure in a local file named `terraform.tfstate`. This backend is simple to use and requires no additional setup, making it a good option for getting started with Terraform.

upvoted 1 times

Using the `terraform state rm` command against a resource will destroy it.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

B (100%)

 **333e1ab** 3 weeks ago

**Selected Answer: B**

How does examTopics evaluate correct answers?

upvoted 1 times

 **Tlakmini** 11 months ago

**Selected Answer: B**

it will update state file only

upvoted 1 times

 **FarziWaliMarzi** 1 year, 2 months ago

**Selected Answer: B**

will update state file only

upvoted 1 times

 **AlenKumar** 1 year, 2 months ago

Answer: B

upvoted 1 times

 **bugalter** 1 year, 3 months ago

**Selected Answer: B**

B. False

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

B. False

Using the terraform state rm command does not destroy the actual resource. Instead, it removes the resource's entry from the Terraform state file. As a result, Terraform will no longer manage or track the resource. However, the resource itself will still exist in your infrastructure. If you want to destroy the resource, you should use the terraform destroy command or modify your configuration to remove the resource and run terraform apply.

upvoted 3 times

 **micropbl4** 1 year, 3 months ago

**Selected Answer: B**

Command just remove it from the state file, but didn't touch existing resources

upvoted 1 times

 **SilentMilli** 1 year, 3 months ago

**Selected Answer: B**

The terraform state rm command removes a resource from the Terraform state file without destroying the actual resource in your infrastructure. This command is typically used when you want to delete a resource that was not created using Terraform, or when you want to remove a resource from the Terraform state file for some other reason, such as to re-create the resource using a different configuration.

upvoted 1 times

Which method for sharing Terraform configurations keeps them confidential within your organization, supports Terraform's semantic version constraints, and provides a browsable directory?

- A. Generic git repository
- B. Terraform Cloud/Terraform Enterprise private module registry
- C. Public Terraform Module Registry
- D. Subfolder within a workspace

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **SilentMilli** 1 year, 3 months ago

**Selected Answer: B**

The Terraform Cloud/Terraform Enterprise private module registry allows organizations to store and share Terraform configurations in a private and secure way. This method keeps the configurations confidential within your organization, supports Terraform's semantic version constraint and provides a browsable directory.

upvoted 3 times

You are writing a child Terraform module which provisions an AWS instance. You want to make use of the IP address returned in the root configuration. You name the instance resource "main".

Which of these is the correct way to define the output value using HCL2?

- A. 

```
output "instance_ip_addr" {
 value = aws_instance.main.private_ip
}
```
- B. 

```
output "aws_instance.instance_ip_addr" {
 value = "${main.private_ip}"
}
```
- C. 

```
output "instance_ip_addr" {
 value = "${aws_instance.main.private_ip}"
}
```
- D. 

```
output "instance_ip_addr" {
 return aws_instance.main.private_ip
}
```

**Correct Answer: C**

*Community vote distribution*

A (100%)

✉️  **wirkmood** 2 months ago

**Selected Answer: A**

It's A.

upvoted 1 times

✉️  **oskarq** 1 year, 2 months ago

**Selected Answer: A**

A is the new one but both would work:

```
output "name-rg" {
 value = azurerm_resource_group.example.name
}
```

```
output "name-rg2" {
 value = "${azurerm_resource_group.example.name}"
}
```

Changes to Outputs:

```
+ name-rg = "example-resource-group"
+ name-rg2 = "example-resource-group"
upvoted 2 times
```

✉️  **FarziWaliMarzi** 1 year, 2 months ago

**Selected Answer: A**

A for sure

upvoted 1 times

✉️  **KakashiCopyNinja** 1 year, 3 months ago

**Selected Answer: A**

It is A. See this documentation: <https://developer.hashicorp.com/terraform/language/values/outputs>  
upvoted 1 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer: A**

It's A

upvoted 1 times

✉️  **tbhttp** 1 year, 3 months ago

its should be A.

A output could look like this:

```
output "example_output" {
 value = aws_instance.example.public_ip
 description = "The public IP address of the example instance."
}
```

upvoted 1 times

✉️  **sandyrao** 1 year, 3 months ago

**Selected Answer: A**

answer should be A

<https://developer.hashicorp.com/terraform/language/values/outputs#declaring-an-output-value>

upvoted 2 times

Question #222

Topic 1

How would you refer to the indexing instance from the below configuration?

```
resource "aws_instance" "web" {
 ...
 for_each = {
 "terraform": "value1",
 "resource": "value2",
 }
```

```
 "indexing": "value3",
 "example": "value4",
 }
}
```

- A. aws\_instance[“web”][“indexing”]
- B. aws\_instance.web.indexing
- C. aws\_instance-web[“indexing”]
- D. aws\_instance.web[“indexing”]

**Correct Answer:** D

*Community vote distribution*

D (90%)

10%

 **d9iceguy** 6 months, 2 weeks ago

**Selected Answer: D**

D is correct

[https://developer.hashicorp.com/terraform/language/meta-arguments/for\\_each#referring-to-instances](https://developer.hashicorp.com/terraform/language/meta-arguments/for_each#referring-to-instances)

upvoted 1 times

 **kareem\_ashraf** 7 months, 3 weeks ago

**Selected Answer: D**

Referring to Instances

with multiple instances are prefixed with module.<NAME>[<KEY>] when displayed in plan output and elsewhere in the UI. For a module with count or for\_each, the address will not contain the module index as the module's name suffices to reference the module.

upvoted 2 times

Question #223

Topic 1

Which feature is not included in Terraform Cloud's free tier?

- A. Workspace
- B. Remote state management
- C. Audit logging
- D. Private module registry

**Correct Answer: D**

*Community vote distribution*

C (92%)

8%

 **uax** 8 months, 3 weeks ago

**Selected Answer: C**

As of 10/27/2023 Audit logging and Team management are not included in the free tier.

[https://www.hashicorp.com/products/terraform/pricing?utm\\_source=google&utm\\_channel\\_bucket=paid&utm\\_medium=sem&utm\\_campaign=CLOUD\\_AMER\\_USA\\_ENG\\_BOFU\\_PRACTITIONER\\_SEN\\_ALL\\_TERRAFORM\\_CLD\\_GG\\_BRAND\\_-\\_Obility&utm\\_content=17089930762-133042352541-595455077952&utm\\_offer=signup&gclid=Cj0KCQjwz6ShBhCMARIsAH9A0qXE10su9PWOCTXGMRRoFQXE0njcbf4SkMiMc7mp\\_Ad7RcdJctwaAvOYEALw\\_wcB](https://www.hashicorp.com/products/terraform/pricing?utm_source=google&utm_channel_bucket=paid&utm_medium=sem&utm_campaign=CLOUD_AMER_USA_ENG_BOFU_PRACTITIONER_SEN_ALL_TERRAFORM_CLD_GG_BRAND_-_Obility&utm_content=17089930762-133042352541-595455077952&utm_offer=signup&gclid=Cj0KCQjwz6ShBhCMARIsAH9A0qXE10su9PWOCTXGMRRoFQXE0njcbf4SkMiMc7mp_Ad7RcdJctwaAvOYEALw_wcB)

upvoted 2 times

 **Jhaggar** 1 year, 2 months ago

**Selected Answer: C**

Audit logging is not included in Terraform Cloud's free tier. Terraform Cloud is a hosted service for Terraform that provides remote state management, a private module registry, and collaboration features such as workspaces and version control integration. However, audit logging is not included in the free tier

upvoted 3 times

 **oskarq** 1 year, 2 months ago

**Selected Answer: C**

<https://www.hashicorp.com/products/terraform/pricing>  
Audit logging only for Cloud Business and Enterprise not for Cloud Free  
Cloud Team & Governance in 2023

upvoted 1 times

oussa\_ama 1 year, 3 months ago

Selected Answer: D

D. Private module registry

Terraform Cloud's free tier includes many features, including workspaces for collaboration, remote state management for tracking changes to infrastructure over time, and audit logging for security and compliance purposes. However, the free tier does not include a private module registry, which is a feature available in the paid tiers.

The private module registry feature allows teams to publish and share their own private Terraform modules, which can be used across different workspaces in the organization. This feature is not included in the free tier and is only available in the paid tiers.

Option A, B, and C are incorrect as workspaces, remote state management, and audit logging are all included in Terraform Cloud's free tier.  
upvoted 1 times

Nunyabiznes 1 year, 3 months ago

Nope, I can tell you're using chatGPT, but chatGPT's most updated date is September 2021.

If you go to this link, it clearly says: Free Tier supporting Private Modules.

Answer is: C

Link: [https://www.hashicorp.com/products/terraform/pricing?utm\\_source=google&utm\\_channel\\_bucket=paid&utm\\_medium=sem&utm\\_campaign=CLOUD\\_AMER\\_USA\\_ENG\\_BOFU\\_PRACTITIONER\\_1M\\_A\\_ALL\\_TERRAFORM\\_CLD\\_GG\\_BRAND\\_-\\_Ability&utm\\_content=17089930762-133042352541-595455077952&utm\\_offer=signup&gclid=Cj0KCQjwz6ShBhCMARlIsAH9A0qXE10su9PWOCTXGMRRoFQXE0njcfbf4SkMiMc7mp\\_Ad7RccPiKwaAvOYEALw\\_wcB](https://www.hashicorp.com/products/terraform/pricing?utm_source=google&utm_channel_bucket=paid&utm_medium=sem&utm_campaign=CLOUD_AMER_USA_ENG_BOFU_PRACTITIONER_1M_A_ALL_TERRAFORM_CLD_GG_BRAND_-_Ability&utm_content=17089930762-133042352541-595455077952&utm_offer=signup&gclid=Cj0KCQjwz6ShBhCMARlIsAH9A0qXE10su9PWOCTXGMRRoFQXE0njcfbf4SkMiMc7mp_Ad7RccPiKwaAvOYEALw_wcB)

upvoted 5 times

tbhttp 1 year, 3 months ago

Selected Answer: C

C. Audit logging is not included in Terraform Cloud's free tier.

Terraform Cloud is a hosted service for Terraform that provides remote state management, a private module registry, and collaboration features such as workspaces and version control integration. However, audit logging is not included in the free tier.

Audit logging allows you to track user activity in Terraform Cloud, including changes made to workspaces, modules, and variables. It provides detailed audit trail that can help you troubleshoot issues, monitor security, and meet compliance requirements.

Audit logging is available in Terraform Cloud's paid tiers, which provide additional features and support options. If you require audit logging or other advanced features, you can upgrade to a paid tier or consider using Terraform Enterprise, which provides additional enterprise-grade features and support.

upvoted 4 times

blanco750 1 year, 3 months ago

Selected Answer: C

Audit logging only supported for terraform cloud business. Not even supported in terraform enterprise. I assume not supported in free plan as well.

<https://www.hashicorp.com/blog/hashicorp-terraform-cloud-audit-logging-with-splunk>

<https://developer.hashicorp.com/terraform/cloud-docs/api-docs/audit-trails>

upvoted 2 times

Saransundar 1 year, 3 months ago

Answer D cannot be correct. From terraform document. "Terraform Cloud includes a private registry that is available to all accounts, including free organizations."

upvoted 1 times

Saransundar 1 year, 3 months ago

Answer is B.

Using this link <https://www.hashicorp.com/products/terraform/pricing> go to "Visibility and Optimization". The feature Audit logging is not available for free tier.

upvoted 1 times

Question #224

Topic 1

When should you run terraform init?

- A. After you run terraform apply for the first time in a new Terraform project and before you run terraform plan
- B. After you run terraform plan for the first time in a new Terraform project and before you run terraform apply
- C. After you start coding a new Terraform project and before you run terraform plan for the first time
- D. Before you start coding a new Terraform project

**Correct Answer:** D

*Community vote distribution*

C (100%)

 **Tlakmini** 11 months ago

**Selected Answer: C**

C.

his is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control.  
upvoted 2 times

 **Jhaggar** 1 year, 2 months ago

**Selected Answer: C**

After you start coding a new Terraform project and before you run Terraform plan for the first time

upvoted 1 times

 **tbhttp** 1 year, 3 months ago

**Selected Answer: C**

C. After you start coding a new Terraform project and before you run terraform plan for the first time

upvoted 2 times

 **micropbl4** 1 year, 3 months ago

**Selected Answer: C**

definitely "C"

unvoted 1 times

Question #225

Topic 1

Terraform configuration (including any module references) can contain only one Terraform provider type.

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

B (100%)

 **Nunyabiznes** Highly Voted 1 year, 3 months ago

**Selected Answer: B**

```
provider "aws" {
region = "us-west-2"
access_key = "ACCESS_KEY"
secret_key = "SECRET_KEY"
}

provider "google" {
project = "my-gcp-project"
credentials = "path/to/google/credentials.json"
}
```

upvoted 5 times

 **camps** Most Recent 1 year, 3 months ago

**Selected Answer: B**

B. False

A Terraform configuration can contain multiple provider types. Providers in Terraform are responsible for understanding API interactions and exposing resources for various cloud platforms and infrastructure services. You can use multiple providers in a single Terraform configuration to manage resources across different platforms or services.

upvoted 2 times

 **tbhttp** 1 year, 3 months ago

**Selected Answer: B**

A Terraform configuration can contain multiple provider blocks, each associated with a different provider type and version.

upvoted 1 times

 **SilentMilli** 1 year, 3 months ago

**Selected Answer: B**

A Terraform configuration can contain multiple provider blocks, each associated with a different provider type and version. This allows you to use multiple providers in a single Terraform configuration, such as using different cloud providers for different resources.

For example, you might use the AWS provider for your EC2 instances and the Azure provider for your virtual machines. You can define each provider block with its own configuration settings, such as access keys or region, and reference the appropriate provider block in the resource blocks that use that provider.

upvoted 2 times

You are making changes to existing Terraform code to add some new infrastructure.

When is the best time to run terraform validate?

- A. After you run terraform plan so you can validate that your state file is consistent with your infrastructure
- B. Before you run terraform plan so you can validate your code syntax
- C. Before you run terraform apply so you can validate your infrastructure changes
- D. After you run terraform apply so you can validate that your infrastructure is reflected in your code

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **camps** 1 year, 3 months ago

**Selected Answer: B**

B. Before you run terraform plan so you can validate your code syntax. Running terraform validate before running terraform plan helps catch syntax errors, missing or incorrect argument declarations, and other issues before any changes are made to the infrastructure.

upvoted 1 times

 **micropbl4** 1 year, 3 months ago

**Selected Answer: B**

There's no need to explain.

You need to make sure that all code described properly

upvoted 2 times

 **SilentMilli** 1 year, 3 months ago

**Selected Answer: B**

B offcourse

upvoted 3 times

How does Terraform manage most dependencies between resources?

- A. By defining dependencies as modules and including them in a particular order
- B. The order that resources appear in Terraform configuration indicates dependencies
- C. Using the depends\_on parameter
- D. Terraform will automatically manage most resource dependencies

**Correct Answer: B**

*Community vote distribution*

D (89%)

11%

 **agg42** 6 months ago

<https://developer.hashicorp.com/terraform/language/resources/behavior#resource-dependencies>

Most resource dependencies are handled automatically. Terraform analyses any expressions within a resource block to find references to other

objects, and treats those references as implicit ordering requirements when creating, updating, or destroying resources. Since most resources with behavioral dependencies on other resources also refer to those resources' data, it's usually not necessary to manually specify dependencies between resources.

upvoted 1 times

✉️  **BaburTurk** 10 months, 3 weeks ago

**Selected Answer: D**

D. Terraform will automatically manage most resource dependencies.

Option A is incorrect because Terraform does not use modules to manage dependencies. Option B is incorrect because the order that resources appear in Terraform configuration does not indicate dependencies.

Option C is incorrect because the depends\_on parameter is used to explicitly declare dependencies, not to automatically manage them.

upvoted 2 times

✉️  **Rajmane** 11 months, 1 week ago

**Selected Answer: C**

It's C depends on

upvoted 1 times

✉️  **Ashwin1011** 11 months, 3 weeks ago

Option D is incorrect because Terraform does not automatically manage all resource dependencies. Some resource dependencies are not visible to Terraform, and you need to explicitly specify them using the depends\_on parameter.

For example, the following Terraform configuration defines an EC2 instance and an IAM role. The EC2 instance depends on the IAM role, but this dependency is not visible to Terraform. You need to explicitly specify the dependency using the depends\_on parameter.

```
resource "aws_instance" "web" {
 depends_on = ["aws_iam_role.role"]
}
```

```
resource "aws_iam_role" "role" {
}
```

If you do not specify the depends\_on parameter, Terraform will try to create the EC2 instance before the IAM role. This will fail because the EC2 instance needs the IAM role to be able to access AWS resources.

In general, it is a good practice to explicitly specify dependencies using the depends\_on parameter. This will help to ensure that your infrastructure is created and updated in the correct order.

upvoted 2 times

✉️  **Freshtimi** 1 year, 2 months ago

D is correct

upvoted 1 times

✉️  **oskarq** 1 year, 2 months ago

**Selected Answer: D**

Only D

upvoted 1 times

✉️  **tbhfp** 1 year, 3 months ago

**Selected Answer: D**

D. Terraform will automatically manage most resource dependencies. Terraform is designed to recognize and manage dependencies between resources automatically, based on the values of their attributes. When a resource references another resource's attributes, Terraform understands the dependency and will create, update, or destroy resources in the correct order to satisfy those dependencies. However, in some cases where explicit dependency management is needed, you can use the "depends\_on" parameter to define additional dependencies.

upvoted 4 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer: D**

D. Terraform will automatically manage most resource dependencies

Terraform is designed to automatically manage most resource dependencies by analyzing the relationships between resources based on their input and output variables. When Terraform detects that a resource relies on another resource's output or attribute, it creates an implicit dependency and ensures that resources are created, updated, or destroyed in the correct order.

upvoted 1 times

✉️  **Witwický** 1 year, 3 months ago

D for me

upvoted 1 times

Question #228

*Topic 1*

What does running a terraform plan do?

- A. Imports all of your existing cloud provider resources to the state file
- B. Compares the state file to your Terraform code and determines if any changes need to be made
- C. Imports all of your existing cloud provider resources to your Terraform configuration file
- D. Compares your Terraform code and local state file to the remote state file in a cloud provider and determines if any changes need to be made

**Correct Answer:** D

*Community vote distribution*

B (88%)

13%

 **SilentH** 2 months, 3 weeks ago

**Selected Answer: D**

Examtopics, please fire whoever you had answer these questions! Almost every single answer is wrong.  
upvoted 1 times

 **tbhttp** 1 year, 3 months ago

**Selected Answer: B**

B. Compares the state file to your Terraform code and determines if any changes need to be made. The terraform plan command is used to create an execution plan, which shows you the changes that will be made to your infrastructure based on the current Terraform configuration the current state file. It allows you to review the changes before actually applying them, helping you understand the impact of your changes and catch any unintended modifications before they happen.

upvoted 2 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

B. Compares the state file to your Terraform code and determines if any changes need to be made

Running a terraform plan command performs a comparison between your Terraform configuration files and the current state file. It then generates an execution plan that shows the differences between the desired state (defined in the configuration) and the actual state (represented by the state file). This plan outlines the actions (create, update, or delete) that Terraform will take to reconcile the differences and achieve the desired state when the terraform apply command is executed.

upvoted 1 times

 **micropbl4** 1 year, 3 months ago

**Selected Answer: B**

B is correct

upvoted 2 times

 **Ravi528** 1 year, 3 months ago

**Selected Answer: B**

B is correct,

upvoted 2 times

What are some benefits of using Sentinel with Terraform Cloud/Terraform Enterprise? (Choose three.)

- A. Policy-as-code can enforce security best practices
- B. You can restrict specific configurations on resources like "CIDR=0.0.0.0/0" not allowed
- C. You can enforce a list of approved AWS AMIs
- D. Sentinel Policies can be written in HashiCorp Configuration Language (HCL)
- E. You can check out and check in cloud access keys

**Correct Answer:** ABC

*Community vote distribution*

ABC (100%)

✉️  **Oleg\_gol** 1 year, 2 months ago

**Selected Answer:** ABC

ABC - 🤝

upvoted 2 times

✉️  **oskarq** 1 year, 2 months ago

**Selected Answer:** ABC

Sentinel can be written in many languages but preferably domain-specific language (DSL) not HCL.

upvoted 2 times

✉️  **tbhttp** 1 year, 3 months ago

**Selected Answer:** ABC

- A. Policy-as-code can enforce security best practices
- B. You can restrict specific configurations on resources like "CIDR=0.0.0.0/0" not allowed
- C. You can enforce a list of approved AWS AMIs

Sentinel is a policy-as-code framework that integrates with Terraform Cloud and Terraform Enterprise, allowing you to enforce policies on your infrastructure as part of the provisioning process. By using Sentinel, you can enforce security best practices, restrict specific configurations such as disallowing overly permissive CIDR blocks, and maintain a list of approved AWS AMIs, among other things. This helps to ensure that your infrastructure is secure, compliant, and adheres to organizational standards.

upvoted 3 times

✉️  **camps** 1 year, 3 months ago

**Selected Answer:** ABC

Answer: A, B, C. Sentinel is a policy-as-code framework that can be used to enforce best practices and security policies on Terraform configurations. Sentinel can be used to restrict specific configurations on resources, enforce a list of approved AWS AMIs, and much more. Sentinel policies can be written in a variety of languages, including HashiCorp Configuration Language (HCL). However, checking out and checking in cloud access keys is not a feature provided by Sentinel.

upvoted 1 times

✉️  **hahano** 1 year, 3 months ago

**Selected Answer:** ABC

You can't checkout keys, and you can't create custom HCL

upvoted 1 times

You want to share Terraform state with your team, store it securely, and provide state locking.

How would you do this? (Choose three.)

- A. Using the remote Terraform backend with Terraform Cloud / Terraform Enterprise.
- B. Using the local backend.
- C. Using the s3 terraform backend. The dynamodb\_field option is not needed.
- D. Using an s3 terraform backend with an appropriate IAM policy and dynamodb\_field option configured.
- E. Using the consul Terraform backend.

**Correct Answer: ADE**

*Community vote distribution*

ADE (86%)

14%

✉️  **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: ADE**

The correct answers are A, D, and E  
upvoted 1 times

✉️  **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: ADE**

It is not C because without dynamo\_db enabled, it does not provide state locking  
upvoted 1 times

✉️  **tbhttp** 1 year, 3 months ago

**Selected Answer: ADE**

A. Using the remote Terraform backend with Terraform Cloud / Terraform Enterprise.  
D. Using an s3 terraform backend with an appropriate IAM policy and dynamodb\_field option configured

Question #231

Topic 1

From which of these sources can Terraform import modules?

- A. Local path
- B. GitHub Repository
- C. Terraform Module Registry
- D. All of the above

**Correct Answer: D**

*Community vote distribution*

D (100%)

✉️  **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: D**

D- <https://developer.hashicorp.com/terraform/language/modules/sources>  
upvoted 1 times

✉️  **tbhttp** 1 year, 3 months ago

**Selected Answer: D**

D. All of the above

Terraform can import modules from a variety of sources, including:

- A. Local path: You can reference a local directory containing your module's Terraform files.
- B. GitHub Repository: You can reference a GitHub repository, specifying the repository URL.
- C. Terraform Module Registry: You can import modules directly from the public Terraform Registry or from private module registries in Terraform Cloud or Terraform Enterprise.

These options make it easy to reuse and share modules across different projects and teams.

upvoted 1 times

✉️  **Ravi528** 1 year, 3 months ago

**Selected Answer: D**

D is correct, all of the above  
upvoted 2 times

Question #232

Topic 1

How would you output returned values from a child module?

- A. Declare the output in the root configuration
- B. Declare the output in the child module
- C. Declare the output in both the root and child module
- D. None of the above

**Correct Answer: C**

*Community vote distribution*

C (58%)

B (42%)

 **tbhttp** Highly Voted 1 year, 3 months ago

**Selected Answer: C**

- C. Declare the output in both the root and child module

To output returned values from a child module, you need to declare the output in both the child and root module.

In the child module, you need to declare an output block, which defines the values that the child module will return.

In the root module, you will reference the output values from the child module using the syntax `module.<module_name>.<output_name>`. You can then declare an output block in the root module to display the values or use them elsewhere in the root module configuration.

upvoted 11 times

 **Oleg\_gol** 1 year, 2 months ago

chatGPT? )

In a parent module, outputs of child modules are available in expressions as `module.<MODULE NAME>.<OUTPUT NAME>`. For example, if a child module named `web_server` declared an output named `instance_ip_addr`, you could access that value as

<https://developer.hashicorp.com/terraform/language/values/outputs>

upvoted 3 times

 **anubha.agrahari** Most Recent 3 months, 1 week ago

B

In a parent module, outputs of child modules are available in expressions as `module.<MODULE NAME>.<OUTPUT NAME>`. For example, if a child module named `web_server` declared an output named `instance_ip_addr`, you could access that value as

`module.web_server.instance_ip_addr`

upvoted 2 times

 **kareem\_ashraf** 7 months, 2 weeks ago

**Selected Answer: B**

you don't need to declare output variable in root it work with child fine

upvoted 1 times

 **Simplon** 9 months ago

**Selected Answer: C**

Just tested it.

If I don't declare an output block in my root module referring to my child module's output , I get no returned output on the CLI.

upvoted 3 times

 **enc\_0343** 9 months, 2 weeks ago

**Selected Answer: C**

You can use `outputs.tf` in the child module and the output block in the root module.

upvoted 1 times

 **Pikopo** 9 months, 3 weeks ago

B is the correct answer in my opinion

upvoted 1 times

✉️  **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: B**

b - <https://developer.hashicorp.com/terraform/language/values/outputs>

upvoted 1 times

✉️  **Misiek** 10 months, 1 week ago

Accessing Child Module Outputs

In a parent module, outputs of child modules are available in expressions as module.<MODULE NAME>.<OUTPUT NAME>. For example, child module named web\_server declared an output named instance\_ip\_addr, you could access that value as module.web\_server.instance\_ip\_addr.

upvoted 1 times

✉️  **oskarq** 1 year, 2 months ago

**Selected Answer: B**

How would you output returned values from a child module?

- A. Declare the output in the root configuration
- B. Declare the output in the child module
- C. Declare the output in both the root and child module
- D. None of the above

upvoted 1 times

✉️  **AlenKumar** 1 year, 2 months ago

Answer B

upvoted 1 times

✉️  **ArnaldoW** 1 year, 3 months ago

**Selected Answer: B**

In Terraform, it is not necessary to declare an output in both the root and child module. The output declared in the child module will be available to the root module, so you only need to declare an output in the root module if you want to expose that value to the outside world or use it in another module.

If you declare an output in both the root and child module with the same name, Terraform will use the output from the root module as the final output. This can be useful if you want to override the output value of the child module with a different value in the root module.

However, it is generally a good practice to avoid duplicate output names between modules to prevent confusion and ensure that the output values are clearly defined and organized.

In summary, it is not necessary to declare an output in both the root and child module, but if you do, the output from the root module will override the output from the child module. It is recommended to use unique output names to avoid confusion and ensure clear organization of output values.

upvoted 4 times

✉️  **Nunyabiznes** 1 year, 3 months ago

**Selected Answer: B**

```
child module:
output "example_output" {
value = "some value"
}
```

```
PARENT Module:
module "example_module" {
source = "./example_module"
}
```

```
output "example_output_from_child" {
value = module.example_module.example_output
}
```

upvoted 2 times

✉️  **Nunyabiznes** 1 year, 3 months ago

The clue here is "returned value" from child module. So the value is already declared in child module, so you just need to call it in Parent

upvoted 4 times

 **camps** 1 year, 3 months ago

**Selected Answer: B**

B. Declare the output in the child module

When you want to output returned values from a child module, you should declare the output in the child module itself. Outputs in Terraform (other Infrastructure as Code tools) are used to expose certain values or results from one module, which can then be consumed by other modules or scripts.

upvoted 2 times

Question #233

*Topic 1*

You have decided to create a new Terraform workspace to deploy a development environment.

What is different about this workspace?

- A. It has its own state file
- B. It pulls in a different `terraform.tfvars` file
- C. It uses a different branch of code
- D. It uses a different backend

**Correct Answer: A**

*Community vote distribution*

A (86%)

14%

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: A**

- A. It has its own state file
- upvoted 1 times

 **styro** 1 year, 3 months ago

**Selected Answer: A**

- A. It has its own state file

When you create a new Terraform workspace, it has its own separate state file. This allows you to manage and deploy multiple environments' different configurations without interfering with each other. Each workspace's state file is isolated from the others, ensuring that changes in one environment don't affect the resources in another environment.

While it is possible for different workspaces to use different `terraform.tfvars` files, branches of code, or even different backends, these differences are not inherently tied to the creation of a new workspace. These variations would depend on how you choose to structure and manage your Terraform configurations for different environments.

upvoted 3 times

 **tbhttp** 1 year, 3 months ago

**Selected Answer: A**

- A. It has its own state file

When you create a new Terraform workspace, it will have its own state file. This separation of state files allows you to manage multiple environments, such as development, staging, and production, without their resources and configurations conflicting with each other. Each workspace will maintain its own state, so you can apply changes to one environment without affecting the others. The other options mentioned (B, C, and D) are not inherently different in a new workspace, but can be configured as needed for the specific environment.

upvoted 1 times

 **camps** 1 year, 3 months ago

**Selected Answer: A**

- A. It has its own state file

When you create a new Terraform workspace, it gets its own state file. Terraform workspaces are designed to manage multiple separate instances of your infrastructure with the same configuration. Each workspace has its own separate state file to track the resources and their states within that specific environment (e.g., development, staging, or production).

upvoted 1 times

 **hahano** 1 year, 3 months ago

**Selected Answer: B**

- You can use workspaces for different environments, which would require different vars

upvoted 1 times

Any user can publish modules to the public Terraform Module Registry.

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Jhaggar** 1 year, 2 months ago

**Selected Answer: A**

any individual

upvoted 1 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: A**

A. - <https://developer.hashicorp.com/terraform/registry/modules/publish>

upvoted 1 times

 **hedykim** 1 year, 2 months ago

**Selected Answer: A**

YES TRUE

upvoted 1 times

 **tbhttp** 1 year, 3 months ago

**Selected Answer: A**

A. True

Any user can publish modules to the public Terraform Module Registry. By publishing a module to the public registry, users can share their infrastructure code with the community and contribute to best practices. The public Terraform Module Registry contains a wide range of modules created by the community and HashiCorp, which can be used to simplify Terraform configurations and promote code reusability.

upvoted 2 times

 **hahano** 1 year, 3 months ago

**Selected Answer: A**

A, yes you can

upvoted 1 times

Which of these commands makes your code more human readable?

A. terraform validate

B. terraform output

C. terraform plan

D. terraform fmt

**Correct Answer: A**

### Community vote distribution

D (88%)

13%

✉️ **Pikopo** 9 months, 3 weeks ago

D is the correct answer

upvoted 1 times

✉️ **Jhaggar** 1 year, 2 months ago

**Selected Answer: D**

The correct answer is D. `terraform fmt` is used to format the Terraform configuration files, making them more human-readable and consistent. This command applies a standard formatting style to your code, which helps to reduce errors and makes it easier to read and maintain.

`terraform validate` is used to check the syntax and validate the Terraform code for errors.

`terraform plan` is used to generate an execution plan that describes what Terraform will do when you apply the configuration.

`terraform output` is used to retrieve the values of output variables defined in your Terraform configuration.

upvoted 2 times

✉️ **FawadK** 1 year, 2 months ago

**Selected Answer: D**

It's D. `terraform fmt`

upvoted 2 times

✉️ **Nobbie** 1 year, 2 months ago

It should be B Terraform Output

<https://developer.hashicorp.com/terraform/cli/commands/show#:~:text=The%20terraform%20show%20command%20is,state%20as%20Terraform%20sees%20it>.

upvoted 2 times

✉️ **FawadK** 1 year, 2 months ago

Wrong. It cannot be B.

The `terraform output` command is used to extract the value of an output variable from the state file.

upvoted 1 times

✉️ **Oleg.gol** 1 year, 2 months ago

**Selected Answer: D**

D. `terraform fmt`

upvoted 1 times

✉️ **Mantis** 1 year, 2 months ago

**Selected Answer: A**

Terraform fmt: <https://developer.hashicorp.com/terraform/cli/commands/fmt>

upvoted 1 times

✉️ **MarshalLaw** 1 year, 2 months ago

**Selected Answer: D**

I would say D since it formats your code in a cleaner way to read code.

upvoted 2 times

Infrastructure as Code (IaC) can be stored in a version control system along with application code.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Jhaggar** 1 year, 2 months ago

**Selected Answer: A**

100% true

upvoted 1 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: A**

A. True

upvoted 2 times

Select the command that doesn't cause Terraform to refresh its state.

- A. terraform apply
- B. terraform destroy
- C. terraform plan
- D. terraform state list

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **Blitz123** 5 months, 2 weeks ago

again a poorly framed question "Terraform to refresh its state" the obvious answer is "D" but "terraform plan" command does not refresh it's states. it reads state file and it compares the configuration against the current state stored in the Terraform state file. It identifies the difference between what is declared in the configuration and what currently exists in the infrastructure. but it in any shape of form it does not refresh the state these kind of questions confuses people. terraform plan is a read-only command if we see from state file point of view, but there are cer aspects, such as we not be running terraform plan when there is a state lock.

upvoted 1 times

 **Spandrop** 7 months ago

why terraform plan would refresh the state file?

upvoted 2 times

 **Spandrop** 7 months ago

got it, it is not about the state file. This kind of question is not about technical skill, but reading skill instead.

upvoted 3 times

 **vindi135** 7 months, 3 weeks ago

**Selected Answer: D**

...or C are both not updating the state.

upvoted 1 times

 **Pete987** 10 months, 3 weeks ago

sorry, my bad - didn't read the question well. Answer is D

upvoted 1 times

 **Pete987** 10 months, 3 weeks ago

A: terraform apply

upvoted 1 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: D**

D. terraform state list

upvoted 3 times

Sentinel policy-as-code is available in Terraform Enterprise.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

A (100%)

 **Ni33** 1 year, 2 months ago

**Selected Answer: A**

A is the correct answer  
upvoted 1 times

 **FawadK** 1 year, 2 months ago

**Selected Answer: A**

Correct answer is A  
upvoted 1 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: A**

A. True  
upvoted 1 times

 **Mantis** 1 year, 2 months ago

**Selected Answer: A**

<https://docs.hashicorp.com/sentinel/terraform>  
upvoted 3 times

Before you can use Terraform's remote backend, you must first execute terraform init.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Jhaggar** 1 year, 2 months ago

**Selected Answer: A**

This command initializes a working directory containing Terraform configuration files, downloads provider plugins and sets up the backend for state storage. If the configuration files have changed, terraform init will also perform an update of the required provider plugins.

upvoted 2 times

 **zanhsieh** 1 year, 2 months ago

NEW QUESTION -

While calling terraform validate, it will call the provider APIs to validate the code.

- A. True
- B. False

I think it shall be A, although I got it wrong in the exam.

upvoted 2 times

 **Foram31** 1 year ago

terraform validate command validates the configuration files in a directory, referring only to the configuration and not accessing any remote services such as remote state, provider APIs, etc.

upvoted 4 times

 **FawadK** 1 year, 2 months ago

**Selected Answer: A**

It's A. True

upvoted 1 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: A**

A. True

upvoted 1 times

Which two steps are required to provision new infrastructure in the Terraform workflow? (Choose two.)

- A. Plan
- B. Apply
- C. Import
- D. Init
- E. Validate

**Correct Answer: BD***Community vote distribution*

BD (74%)

AB (26%)

  **Simplon** Highly Voted 9 months ago**Selected Answer: BD**

Plan is recommended but not a requirement, whereas init is mandatory and apply is necessary to provision infrastructure.

upvoted 6 times

  **TheShantyman** Most Recent 1 month, 2 weeks ago**Selected Answer: BD**

BD is correct. The apply will automatically execute the Plan, so A is not explicitly required.

upvoted 1 times

  **Jas14** 9 months, 2 weeks ago

people voting for A, seriously !!

upvoted 2 times

  **Pikopo** 9 months, 3 weeks ago

BD is correct

upvoted 2 times

  **seifskl** 11 months, 3 weeks ago**Selected Answer: AB**

<https://developer.hashicorp.com/terraform/intro/core-workflow>

The core Terraform workflow has three steps:

Write - Author infrastructure as code.

Plan - Preview changes before applying.

Apply - Provision reproducible infrastructure.

upvoted 1 times

  **ShakDaddy** 11 months, 2 weeks ago

You do not need to use "plan" independently since "apply" automatically runs plan as well.

You can test it yourself by writing a configuration, running "init" and then "apply".

upvoted 5 times

  **Jhaggar** 1 year, 2 months ago**Selected Answer: BD**

First init and then apply

upvoted 2 times

  **Pachecohete** 1 year, 2 months ago**Selected Answer: BD**

BD, plan is not a requirement.

upvoted 3 times

  **tabkar** 1 year, 2 months ago**Selected Answer: BD**

I believe in should be B & D

upvoted 2 times

  **Oleg\_gol** 1 year, 2 months ago**Selected Answer: AB**

A. Plan and B. Apply

<https://developer.hashicorp.com/terraform/intro/core-workflow>

upvoted 4 times

  **[Removed]** 1 year ago

I agree with this.

upvoted 2 times

Question #241

Topic 1

You are working on some new application features and you want to spin up a copy of your production deployment to perform some quick tests. In order to avoid having to configure a new state backend, what open source Terraform feature would allow you create multiple states but still be associated with your current code?

- A. Terraform data sources
- B. Terraform local values
- C. Terraform modules
- D. Terraform workspaces
- E. None of the above

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Stargazer11** 8 months, 1 week ago

D. Terraform workspaces  
upvoted 2 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: D**  
D. Terraform workspaces  
upvoted 4 times

Which provisioner invokes a process on the machine running Terraform?

- A. remote-exec
- B. file
- C. local-exec
- D. null-exec

**Correct Answer:** C

*Community vote distribution*

C (80%)

A (20%)

 **trextor** 2 months ago

C. local-exec

The local-exec provisioner is used to execute a command or script locally on the machine running Terraform. It's often used for tasks such as running scripts after resource creation or performing local setup/configuration tasks.

upvoted 1 times

 **enry99ita** 7 months, 1 week ago

**Selected Answer: A**

local-exec provisioner is execute on the local machine.

upvoted 1 times

 **enry99ita** 7 months ago

vote C

upvoted 2 times

 **tabkar** 1 year, 2 months ago

**Selected Answer: C**

<https://developer.hashicorp.com/terraform/language/resources/provisioners/local-exec>

upvoted 3 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: C**

C. local-exec

upvoted 1 times

\_\_\_\_\_ backends support state locking.

- A. Some
- B. No
- C. Only local
- D. All

**Correct Answer: A**

*Community vote distribution*

A (89%)

11%

 **Oleg\_gol** Highly Voted 1 year, 2 months ago

**Selected Answer: A**

Not all backends support locking. The documentation for each backend includes details about whether it supports locking or not.  
<https://developer.hashicorp.com/terraform/language/state/backends>

upvoted 8 times

 **ravi135** Most Recent 5 months ago

A is right

upvoted 1 times

 **meetplanet** 1 year, 2 months ago

**Selected Answer: D**

because it indicates that all backends support state locking. State locking is a mechanism used to control access to shared resources to ensure data consistency and prevent conflicts in concurrent or parallel systems.

upvoted 1 times

 **meetplanet** 1 year, 2 months ago

Correct answer is A. sorry for the confusion.

upvoted 2 times

Which of the following methods, used to provision resources into a public cloud, demonstrates the concept of infrastructure as code?

- A. curl commands manually run from a terminal
- B. A sequence of REST requests you pass to a public cloud API endpoint
- C. A script that contains a series of public cloud CLI commands
- D. A series of commands you enter into a public cloud console

**Correct Answer: C**

*Community vote distribution*

C (50%)

B (50%)

 **8876ca1** 3 weeks, 6 days ago

**Selected Answer: C**

C. A script that contains a series of public cloud CLI commands:

This is correct. A script (e.g., using Bash, PowerShell, or another scripting language) that automates the provisioning of resources aligns with principles of IaC. It is automated, repeatable, and can be version-controlled.

Atte ChatGPT 4o

upvoted 1 times

✉️ **TheShantyman** 1 month, 2 weeks ago

**Selected Answer: C**

C is correct. It is the only option that persists the commands being run so that they can be re-used again later. All other options will need to be manually executed again if you wanted to recreate the infra config.

upvoted 1 times

✉️ **Roman\_Rabodzey** 4 months ago

**Selected Answer: B**

B is correct. For Azure Cloud, you provide an API version of a resource you are going to deploy.

upvoted 2 times

✉️ **iamabhi** 4 months, 1 week ago

Selected Answer: C

The concept of infrastructure as code (IaC) is to define and manage infrastructure using code, rather than manual processes or GUI tools. A script that contains a series of public cloud CLI commands is an example of IaC, because it uses code to provision resources into a public cloud. The other options are not examples of IaC, because they involve manual or interactive actions, such as running curl commands, sending REST requests, or entering commands into a console. Reference= [Introduction to Infrastructure as Code with Terraform] and [Infrastructure as Code with AWS CloudFormation]

upvoted 1 times

✉️ **ae07177** 7 months ago

Correct Answer is B.

Hashicorp Documentation Says it's not Step C, so it must be B.

IaC Makes Infrastructure More Reliable

IaC makes changes idempotent, consistent, repeatable, and predictable. Without IaC, scaling up infrastructure to meet increased demand may require an operator to remotely connect to each machine and then manually provision and configure many servers by executing a series of commands/scripts.

<https://www.hashicorp.com/blog/infrastructure-as-code-in-a-private-or-public-cloud>

upvoted 2 times

✉️ **Tronko86** 8 months, 2 weeks ago

**Selected Answer: C**

The method that demonstrates the concept of infrastructure as code when provisioning resources in a public cloud is:

C. A script that contains a series of public cloud CLI commands

Infrastructure as code (IaC) involves defining and provisioning infrastructure resources using code and scripts. In this case, using a script with commands allows you to automate the provisioning process and manage infrastructure configurations programmatically. It provides the benefit of version control, repeatability, and automation, which are core principles of IaC.

upvoted 2 times

✉️ **Pikopo** 9 months, 3 weeks ago

B is correct in my opinion

upvoted 2 times

✉️ **lotfi50** 11 months, 3 weeks ago

**Selected Answer: C**

C. A script that contains a series of public cloud CLI commands

upvoted 1 times

 **seifskl** 11 months, 3 weeks ago

**Selected Answer: C**

The correct answer is C.

The idea behind Infrastructure as Code (IaC) is to write and execute code to define, deploy, and update infrastructure. If that infrastructure code is written in a high-level language, then it can be version controlled and audited, which has benefits for traditional software development practices, such as code review, continuous integration, and automated testing.

Option C, where a script contains a series of public cloud CLI commands, best fits this concept. This script can be version controlled, reviewed, and executed consistently across environments, adhering to the principles of IaC.

The other options are manual methods that do not adhere to the principles of IaC. They do not support versioning or auditing, are error-prone and can't ensure consistency across multiple deployments.

upvoted 1 times

 **VSMu** 12 months ago

**Selected Answer: C**

I would answer C because script is a code in a file ( like bash script) that can be executed multiple times. It can be pushed to a source control repo and can be versioned. Calling commands directly whether they are through CLI, APIs or through Console cannot be repeated like a script.

upvoted 1 times

 **TafMuko** 12 months ago

**Selected Answer: B**

B makes more sense

upvoted 2 times

 **amoyano** 1 year ago

**Selected Answer: B**

you declare your desired config and then Terraform performs a sequence of API calls to your cloud provider. It's not a script with CLI commands.

upvoted 4 times

 **chaoscreator** 1 month, 2 weeks ago

Wrong. You can use pipelines to execute CLI commands. In fact, you can execute Powershell scripts etc in Terraform itself. The answer can be B because it's not in code. Question is asking about infrastructure as code.

upvoted 1 times

 **atiiii** 1 year ago

Doesn't B make more sense?

upvoted 2 times

 **Olea mol** 1 year, 2 months ago

Question #245

Topic 1

Which of the following should you put into the required\_providers block?

A. version >= 3.1

B. version = ">= 3.1"

C. version ~> 3.1

**Correct Answer: C**

*Community vote distribution*

B (64%)

C (29%)

7%

 **bella** 1 month, 2 weeks ago

well B and C are correct. the question isn't clear about required provider... did they mean terraform version or hashicorp provider

```
terraform {
 required_providers {
 aws = {
 source = "hashicorp/aws"
 version = "~> 5.42"
 }
 }
}
```

```
}
```

required\_version = ">= 1.2.0"  
}

upvoted 1 times

✉  **Alandt** 5 months, 3 weeks ago

**Selected Answer: B**

B is correct, others have wrong syntax  
upvoted 1 times

✉  **Tronko86** 8 months, 2 weeks ago

**Selected Answer: B**

B is the only answer with a correct syntax: version parameter wants a string  
upvoted 2 times

✉  **bajwa360** 8 months, 3 weeks ago

```
terraform {
 required_providers {
 mycloud = {
 source = "mycorp/mycloud"
 version = "~> 1.0"
 }
 }
}
```

upvoted 1 times

✉  **arun00028** 9 months ago

**Selected Answer: C**

C is the answer  
upvoted 1 times

✉  **arun00028** 9 months ago

**Selected Answer: C**

C is correct  
upvoted 1 times

✉  **Sathisgm** 1 year, 2 months ago

I choose B  
upvoted 1 times

✉  **MarshalLaw** 1 year, 2 months ago

**Selected Answer: B**

B since this syntax is correct. If you go back to the previous questions you'll see a screenshot where a relevant question is being asked.  
upvoted 2 times

✉  **sdm13168** 1 year, 2 months ago

**Selected Answer: B**

B  
<https://developer.hashicorp.com/terraform/language/providers/requirements>  
upvoted 4 times

✉  **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: C**

I choose "C"  
upvoted 2 times

✉  **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: A**

A and C - correct  
upvoted 1 times

When should you write Terraform configuration files for existing infrastructure that you want to start managing with Terraform?

- A. Before you run terraform import
- B. You can import infrastructure without corresponding Terraform code
- C. Terraform will generate the corresponding configuration files for you
- D. After you run terraform import

**Correct Answer: A**

*Community vote distribution*

A (72%)

D (28%)

 **Ni33** Highly Voted 1 year, 2 months ago

**Selected Answer: A**

A is the correct answer !

upvoted 13 times

 **blop213** Most Recent 2 months, 2 weeks ago

If D is right then B is necesarry right, and since its not multiple choice, it can only be A

Right answer : A

upvoted 1 times

 **Tricejer** 5 months, 2 weeks ago

**Selected Answer: A**

A is correct

upvoted 1 times

 **Blitz123** 5 months, 2 weeks ago

as per terraform 1.0 which is the latest version for exam

its B. You can import infrastructure without corresponding Terraform code

upvoted 2 times

 **uax** 8 months, 3 weeks ago

**Selected Answer: A**

When should you write Terraform configuration files for existing infrastructure that you want to start managing with Terraform?

A. Before you run terraform import.

upvoted 1 times

 **AdriBFK** 10 months ago

**Selected Answer: A**

Tested

upvoted 1 times

 **arunrkaushik** 11 months ago

You cannot import the resource unless the resource's basic block is not coded in the .tf file. Hence this is the 1st step. Once a basic block is written, you import the resources and enhance the code using the state details.

<b> Before you run terraform import </b>

upvoted 2 times

 **seifskl** 11 months, 3 weeks ago

**Selected Answer: A**

Please refer to this link <https://developer.hashicorp.com/terraform/cli/import/usage>

It is said:

"To import a resource, first write a resource block for it in your configuration, establishing the name by which it will be known to Terraform

..."

Now terraform import can be run to attach an existing instance to this resource configuration"

upvoted 2 times

 **lukacs16** 1 year ago

**Selected Answer: A**

"To import a resource, first write a resource block for it in your configuration, establishing the name by which it will be known to Terraform"

Answer is A.

<https://developer.hashicorp.com/terraform/cli/import/usage>

upvoted 2 times

 **Foram31** 1 year ago

**Selected Answer: D**

1. Run terraform import to bring the existing resources under Terraform management. This command allows you to import the existing resources into the Terraform state without the corresponding Terraform configuration files.

2. After importing the resources, you should write the Terraform configuration files to define and manage the imported resources. These configuration files specify the desired state of the infrastructure and allow you to make changes to it in a controlled manner.

upvoted 2 times

 **yubac** 1 year, 1 month ago

**Selected Answer: A**

Considering an existing infrastructure on a cloud provider, in order to manage these resources on terraform you need:

1. foreach resource, add at least the resources definition using resource\_type.resource\_name {}
2. foreach resource, use terraform import resource\_type.resource\_name <id>

upvoted 1 times

 **Sydurrahman12** 1 year, 1 month ago

which one right A or D?

upvoted 1 times

 **sdm13168** 1 year, 2 months ago

**Selected Answer: A**

<https://spacelift.io/blog/importing-existing-infrastructure-into-terraform>

upvoted 2 times

 **Sathisgm** 1 year, 2 months ago

Answer is D

upvoted 2 times

 **MarshallLaw** 1 year, 2 months ago

**Selected Answer: D**

D:

You can use the import command to migrate existing resources into your Terraform state file. The import command does not currently generate the configuration for the imported resource, so you must write the corresponding configuration block to map the imported resource to it.

Importing infrastructure involves five steps:

- 1 Identify the existing infrastructure you will import.
- 2 Import infrastructure into your Terraform state file.
- 3 Write Terraform configuration that matches that infrastructure.
- 4 Review the Terraform plan to ensure the configuration matches the expected state and infrastructure.
- 5 Apply the configuration to update your Terraform state.

Source: <https://developer.hashicorp.com/terraform/tutorials/state/state-import>

upvoted 4 times

 **[Removed]** 1 year ago

Answer is A.

Source: <https://developer.hashicorp.com/terraform/tutorials/state/state-import>. Has been updated with new steps for Terraform v1.5

Question #247

*Topic 1*

Which command should you run to check if all code in a Terraform configuration that references multiple modules is properly formatted without making changes?

- A. `terraform fmt -write=false`
- B. `terraform fmt -list -recursive`
- C. `terraform fmt -check -recursive`
- D. `terraform fmt -check`

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **seifskl** 11 months, 3 weeks ago

**Selected Answer: C**

C. `terraform fmt -check -recursive`

The `terraform fmt -check -recursive` command checks if the files are formatted according to the Terraform language style conventions. The `-check` option will make the command return a non-zero exit code if any of the files are not properly formatted, and the `-recursive` option instructs it to go into the sub-directories as well, which is useful when you have a Terraform configuration that references multiple modules.

upvoted 4 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: C**

C. `terraform fmt -check -recursive`

upvoted 1 times

What features stops multiple users from operating on the Terraform state at the same time?

- A. Provider constraints
- B. Remote backends
- C. State locking
- D. Version control

**Correct Answer:** C

*Community vote distribution*

C (100%)

 LemonadeSoftware 8 months ago

- C. State locking

State locking in Terraform prevents multiple users or processes from simultaneously modifying the Terraform state file (terraform.tfstate). It ensures that only one operation (e.g., terraform apply or terraform destroy) can modify the state at a time, preventing conflicts in collaborative automated environments.

upvoted 1 times

 Oleg\_gol 1 year, 2 months ago

**Selected Answer: C**

- C. State locking

upvoted 4 times

You are creating a reusable Terraform configuration and want to include a billing\_dept tag so your Finance team can track team-specific spending on resources. Which of the following billing\_dept variable declarations will allow you to do this?

- A. 

```
variable "billing_dept" {
 optional = true
}
```
- B. 

```
variable "billing_dept" {
 type = optional(string)
}
```
- C. 

```
variable "billing_dept" {
 default = ""
}
```
- D. 

```
variable "billing_dept" {
 type = default
}
```

**Correct Answer:** C

*Community vote distribution*

C (67%)

B (33%)

👤 **Simplon** 9 months ago

**Selected Answer: C**

Tested. All other answers returned an error.javascript:void(0)  
upvoted 1 times

👤 **Simplon** 9 months ago

\*Tested. All other answers returned an error. (the javascript:void(0) is a typing mistake on my part).  
upvoted 1 times

👤 **halfway** 1 year, 1 month ago

**Selected Answer: C**

It is C, optional, any data type is acceptable.  
upvoted 1 times

👤 **sdm13168** 1 year, 2 months ago

**Selected Answer: C**

tested, got error for A. But C verified  
upvoted 4 times

👤 **Jhaggar** 1 year, 2 months ago

**Selected Answer: B**

type = optional(string)  
upvoted 1 times

👤 **Ni33** 1 year, 2 months ago

C is the correct answer. Tested and verified.  
upvoted 2 times

👤 **Ni33** 1 year, 2 months ago

**Selected Answer: C**

C is the correct option. All the arguments of variables are optional. A - optional is not a valid argument. B- optional(string) is not a valid argument value. D - default is not a valid argument value.  
upvoted 2 times

👤 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: B**

variable "billing\_dept "{  
type = optional(string)  
}  
upvoted 3 times

👤 **szyd3rca** 1 year, 2 months ago

| Error: Invalid type specification  
|  
| on main.tf line 15, in variable "billing\_dept":  
| 15: type = optional(string)  
|  
| Keyword "optional" is valid only as a modifier for object type attributes.  
upvoted 2 times

Which of these are secure options for storing secrets for connecting to a Terraform remote backend? (Choose two.)

- A. Inside the backend block within the Terraform configuration
- B. Defined in Environment variables
- C. Defined in a connection configuration outside of Terraform
- D. A variable file

**Correct Answer:** BC

*Community vote distribution*

BC (88%)

13%

 **Oleg\_gol** Highly Voted 1 year, 2 months ago

**Selected Answer:** BC

BC - 

upvoted 7 times

 **dankositzke** Most Recent 4 months, 3 weeks ago

**Selected Answer:** BD

BD.

(B) clearly yes

Now it comes down to between (C) and (D).

(C) would not be appropriate because connection configurations deal with connections which are not directly relevant to the question.

(D) is more relevant and correct because you can pass in your variable file on using the CLI which will allow your secrets to exist outside of the main code

upvoted 1 times

 **dev\_maftuna** 7 months, 2 weeks ago

The secure options for storing secrets for connecting to a Terraform remote backend are:

- B. Defined in Environment variables
- D. A variable file

Environment variables provide a secure way to store sensitive information without exposing it directly in the code. Using a variable file allows separate storage of sensitive data, which can be managed and secured independently from the Terraform configuration itself. Storing secrets inside the backend block or in a connection configuration outside of Terraform might expose sensitive information within the configuration file external settings, which could pose a security risk.

upvoted 3 times

You want to define a single input variable to capture configuration values for a server. The values must represent memory as a number, and the server name as a string.

Which variable type could you use for this input?

- A. List
- B. Object
- C. Map
- D. Terraform does not support complex input variables of different types

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **seifskl** 11 months, 3 weeks ago

**Selected Answer: B**

B. Object

The Object type in Terraform allows you to create complex input variables that contain more than one value and can be of different types.

```
variable "server_config" {
 type = object({
 memory = number
 name = string
 })
 description = "Server configuration values"
}
```

In this example, `server_config` is an object that expects two attributes: `memory` (a number) and `name` (a string).

upvoted 2 times

 **AWS\_cert2023** 1 year ago

Why not B not C?

<https://developer.hashicorp.com/terraform/language/expressions/type-constraints>  
`map(...)`: a collection of values where each is identified by a string label.  
`object(..)`: a collection of named attributes that each have their own type.

Values of map is string, values of object is any type.

upvoted 1 times

 **Sathisgm** 1 year, 2 months ago

I choose B

upvoted 1 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: B**

B  
<https://developer.hashicorp.com/terraform/language/values/variables>  
upvoted 3 times

What does Terraform not reference when running a `terraform apply -refresh-only`?

- A. Credentials
- B. State file
- C. Terraform resource definitions in configuration files
- D. Cloud provider

**Correct Answer: C**

*Community vote distribution*

|         |         |    |
|---------|---------|----|
| C (67%) | A (20%) | 7% |
|---------|---------|----|

 **kiran15789** Highly Voted 1 year, 2 months ago

**Selected Answer: C**

The `-refresh-only` flag tells Terraform to skip the resource creation and update steps and only update the state file with the current state of resources in the cloud provider. This means that Terraform will not reference the resource definitions in the configuration files, since it is not applying any changes to the cloud provider based on those definitions.

upvoted 9 times

 **Alandt** Most Recent 5 months, 3 weeks ago

**Selected Answer: C**

I'll go with C

upvoted 1 times

Question #253

Topic 1

Multiple team members are collaborating on infrastructure using Terraform and want to format their Terraform code following standard Terraform-style convention. How could they automatically ensure the code satisfies conventions?

- A. Run the `terraform fmt` command during the code linting phase of your CI/CD process
- B. Manually apply two spaces indentation and align equal sign `=` characters in every Terraform file (`*.tf`)
- C. Run the `terraform validate` command prior to executing `terraform plan` or `terraform apply`

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Oleg\_gol** Highly Voted 1 year, 2 months ago

**Selected Answer: A**

By running `terraform fmt` during the code linting phase of your CI/CD process, you can automatically ensure that all Terraform files are formatted according to the standard conventions before any code is merged into the main branch. This helps to maintain a consistent code style and reduces the likelihood of errors caused by inconsistent formatting.

upvoted 8 times

 **Paul\_white** 11 months, 2 weeks ago

A. Credentials

When running `terraform apply -refresh-only`, Terraform does not reference credentials. The `-refresh-only` flag is used to perform a refresh of the state without making any changes to the actual resources in the cloud provider. During this operation, Terraform retrieves the current state of resources from the cloud provider, updates the state file with the latest information, but it does not perform any actions that require credential (e.g., creating, modifying, or deleting resources).

The `-refresh-only` flag is useful when you want to synchronize Terraform's understanding of the resources with the real state in the cloud provider without performing any destructive actions. It allows you to update the state file with the latest resource information without making any changes to the infrastructure.

upvoted 2 times

 **Oleg\_gol** 1 year, 2 months ago

**Selected Answer: D**

<https://developer.hashicorp.com/terraform/tutorials/state/refresh>

upvoted 1 times

When using a remote backend or Terraform Cloud integration, where does Terraform save resource state?

- A. On the disk
- B. In memory
- C. In an environment variable
- D. In the remote backend or Terraform Cloud

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **Foram31** 1 year ago

**Selected Answer: D**

D. In the remote backend or Terraform Cloud

With a remote backend configuration or Terraform Cloud integration, the state data, which includes information about the resources managed by Terraform, is stored and managed in the remote backend or Terraform Cloud. This allows for centralized state management, collaboration, and concurrent access to the state file by multiple users or team members.

upvoted 3 times

 **Fyssy** 1 year ago

**Selected Answer: D**

D is correct

upvoted 1 times

In Terraform HCL, an object type of object({ name=string, age=number }) would match this value:

- A. 

```
{
 name = "John"
 age = fifty two
}
```
- B. 

```
{
 name = "John"
 age = 52
}
```
- C. 

```
{
 name = John
 age = fifty two
}
```

D.

```
{
 name = John
 age = 52
}
```

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **March2023** Highly Voted 1 year, 1 month ago

**Selected Answer: B**

B is correct

upvoted 6 times

 **nileeka97** Most Recent 5 months, 4 weeks ago

**Selected Answer: B**

B for sure

upvoted 1 times

 **LemonadeSoftware** 8 months ago

B

The name (string) must be written in quotes. The age (number) should not be in quotes and must be written with numeric characters

upvoted 2 times

You add a new resource to an existing Terraform configuration, but do not update the version constraint in the configuration. The existing and new resources use the same provider. The working directory contains a .terraform-lock.hcl file.

How will Terraform choose which version of the provider to use?

- A. Terraform will use the latest version of the provider for the new resource and the version recorded in the lock file to manage existing resources
- B. Terraform will use the version recorded in your lock file
- C. Terraform will check your state file to determine the provider version to use
- D. Terraform will use the latest version of the provider available at the time you provision your new resource

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **DevoteamAnalytix** 1 year ago

**Selected Answer: B**

<https://developer.hashicorp.com/terraform/language/files/dependency-lock#dependency-installation-behavior>  
upvoted 2 times

 **Foram31** 1 year ago

**Selected Answer: B**

When Terraform encounters a new resource that uses the same provider, it will consult the .terraform-lock.hcl file to determine the version of the provider to use for that resource. This ensures consistency in the versions used across resources in the configuration and helps maintain reproducibility.

upvoted 1 times

 **simoziyadi** 1 year, 1 month ago

B is correct

upvoted 2 times

You must use different Terraform commands depending on the cloud provider you use.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **seifskl** 11 months, 3 weeks ago

**Selected Answer: B**

B. False

the commands to use to interact with Terraform itself (such as terraform apply, terraform plan, and so on) remain the same regardless of the c provider.

upvoted 2 times

 **AWS\_cert2023** 1 year ago

The answer B.

upvoted 3 times

Define the purpose of state in Terraform.

- A. State stores variables and lets you quickly reuse existing code
- B. State lets you enforce resource configurations that relate to compliance policies
- C. State codifies the dependencies of related resources
- D. State maps real world resources to your configuration and keeps track of metadata

**Correct Answer:** D

*Community vote distribution*

D (100%)

✉️  **DevoteamAnalytix** 1 year ago

**Selected Answer: D**

<https://developer.hashicorp.com/terraform/language/state/purpose#mapping-to-the-real-world>  
<https://developer.hashicorp.com/terraform/language/state/purpose#metadata>

upvoted 2 times

✉️  **March2023** 1 year ago

**Selected Answer: D**

meta data

upvoted 1 times

✉️  **AWS\_cert2023** 1 year ago

The answer D.

upvoted 2 times

Which of these actions will prevent two Terraform runs from changing the same state file at the same time?

- A. Refresh the state after running Terraform
- B. Delete the state before running Terraform
- C. Configure state locking for your state backend
- D. Run Terraform with parallelism set to 1

**Correct Answer:** C

*Community vote distribution*

C (100%)

✉️  DevoteamAnalytix 1 year ago

**Selected Answer: C**

"If supported by your backend, Terraform will lock your state for all operations that could write state. This prevents others from acquiring the lock and potentially corrupting your state."

<https://developer.hashicorp.com/terraform/language/state/locking>

upvoted 2 times

While attempting to deploy resources into your cloud provider using Terraform, you begin to see some odd behavior and experience slow responses. In order to troubleshoot you decide to turn on Terraform debugging. Which environment variables must be configured to make Terraform's logging more verbose?

- A. TF\_LOG\_PATH
- B. TF\_VAR\_log\_level
- C. TF\_LOG
- D. TF\_VAR\_log\_path

**Correct Answer:** C

*Community vote distribution*

C (100%)

✉️  DevoteamAnalytix 1 year ago

**Selected Answer: C**

"You can set TF\_LOG to one of the log levels (in order of decreasing verbosity) TRACE, DEBUG, INFO, WARN or ERROR to change the verbosity of the logs."

<https://developer.hashicorp.com/terraform/internals/debugging>

upvoted 2 times

The Terraform binary version and provider versions must match each other in a single configuration.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

✉️  **LemonadeSoftware** 8 months ago

B

Not necessarily. In Terraform, the version of the Terraform binary itself doesn't have to match the versions of the providers being used within a configuration file.

However, it's generally recommended to use compatible versions. Terraform typically manages provider versions independently from its core binary. Each Terraform provider (e.g., AWS, Azure, etc.) has its own versioning that may differ from the version of the Terraform core.

Using compatible versions helps ensure that the features and functionalities expected by your configuration are available and supported by both the Terraform binary and the specified providers. But it's not a strict requirement that they must match exactly.

upvoted 1 times

✉️  **Fyssy** 1 year ago

**Selected Answer: B**

If the versions are incompatible or if a required provider is missing, Terraform will prompt you to update the provider versions or download the necessary provider plugins.

upvoted 3 times

The .terraform.lock.hcl file tracks module versions.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (87%)

13%

✉️  **Fyssy**  1 year ago

**Selected Answer: B**

The .terraform.lock.hcl file is not used to track module versions; it is used to lock the versions of the provider dependencies used by your Terraform configuration.

upvoted 9 times

 **Newuser95** Most Recent 4 months, 3 weeks ago

**Selected Answer: B**

At present, the dependency lock file tracks only provider dependencies. Terraform does not remember version selections for remote modules, and so Terraform will always select the newest available module version that meets the specified version constraints. You can use an exact version constraint to ensure that Terraform will always select the same module version.

<https://developer.hashicorp.com/terraform/language/files/dependency-lock>

upvoted 2 times

 **gold4otas** 6 months, 2 weeks ago

**Selected Answer: B**

The .terraform.lock.hcl file does not track module versions. Instead, it is used to lock the versions of the provider plugins used by Terraform. It helps ensure that the same provider versions are used consistently across different Terraform runs and by different team members.

Module versions, on the other hand, are typically tracked in the versions.tf or required\_version block in the main Terraform configuration file or the root module. The versions of modules are controlled through the Terraform configuration itself, not the .terraform.lock.hcl file.

upvoted 1 times

 **Spandrop** 7 months ago

**Selected Answer: A**

<https://developer.hashicorp.com/terraform/language/files/dependency-lock>

upvoted 1 times

 **enry99ita** 7 months, 1 week ago

**Selected Answer: B**

At present, the dependency lock file tracks only provider dependencies. Terraform does not remember version selections for remote modules, and so Terraform will always select the newest available module version that meets the specified version constraints. You can use an exact version constraint to ensure that Terraform will always select the same module version.

upvoted 1 times

 **PolitoMex** 7 months, 2 weeks ago

**Selected Answer: A**

In the newest documentation .terraform.lock.hcl tracks both modules and providers versions.

upvoted 1 times

 **LemonadeSoftware** 8 months ago

A

The .terraform.lock.hcl file keeps track of the specific versions of modules used in your Terraform configuration. It records the exact versions of modules and their dependencies to ensure that subsequent runs of Terraform use the same versions consistently.

This file helps maintain consistency across different environments and team members working on the same Terraform project by locking the versions of modules, preventing unintentional upgrades or changes to the modules without explicit action.

upvoted 2 times

 **dn\_mohammed\_data** 9 months, 3 weeks ago

**Selected Answer: B**

tracks both providers and modules versions

upvoted 2 times

 **uax** 8 months, 2 weeks ago

"At present, the dependency lock file tracks only provider dependencies. Terraform does not remember version selections for remote modules, and so Terraform will always select the newest available module version that meets the specified version constraints."

<https://developer.hashicorp.com/terraform/language/files/dependency-lock>

upvoted 2 times

 **Mimi666** 9 months, 3 weeks ago

**Selected Answer: B**

At present, the dependency lock file tracks only provider dependencies. Terraform does not remember version selections for remote modules, and so Terraform will always select the newest available module version that meets the specified version constraints.

Fuente: <https://developer.hashicorp.com/terraform/language/files/dependency-lock>

upvoted 4 times

✉️  **BalaGCPArch** 10 months, 3 weeks ago

**Selected Answer: A**

The question does not ask if it's used to track only module version.. the dependency lock is used to track both provider and module version.. Hence I go with Option A

upvoted 1 times

✉️  **akm\_1010** 11 months ago

Question #263

Topic 1

es

You can develop a custom provider to manage its resources using Terraform.

A. True

B. False

**Correct Answer: A**

*Community vote distribution*

A (78%)

B (22%)

✉️  **Newuser95** 4 months, 3 weeks ago

**Selected Answer: A**

<https://www.hashicorp.com/blog/writing-custom-terraform-providers>

upvoted 1 times

✉️  **Spandrop** 7 months ago

**Selected Answer: B**

I would say false.

You can create the provider using Go language for instance, but not Terraform (HCL).

That is pretty clear in this tutorial:

<https://developer.hashicorp.com/terraform/tutorials/providers-plugin-framework/providers-plugin-framework-provider>

You develop everything with Go and then test in Terraform

upvoted 1 times

✉️  **velmidos** 9 months ago

**Selected Answer: A**

Answer is A !

upvoted 4 times

✉️  **Paul\_white** 11 months, 2 weeks ago

ANSWER IS A

upvoted 2 times

✉️  **Sam1101** 11 months, 2 weeks ago

**Selected Answer: B**

Answer is True

upvoted 1 times

✉️  **DevoteamAnalytix** 1 year ago

**Selected Answer: A**

<https://developer.hashicorp.com/terraform/tutorials/providers-plugin-framework/providers-plugin-framework-provider>

upvoted 2 times

✉️  **Rajmane** 11 months, 1 week ago

Right correct answer is A

upvoted 1 times

Question #264

*Topic 1*

Which of these is not a benefit of remote state?

- A. Keeping unencrypted sensitive information off disk
- B. Easily share reusable code modules
- C. Working in a team
- D. Delegate output to other teams

**Correct Answer: B**

*Community vote distribution*

B (50%)

D (36%)

14%

 BaburTurk Highly Voted 10 months, 3 weeks ago

**Selected Answer: D**

The answer is D. Delegate output to other teams.

Remote state is a feature of Terraform that allows you to store the state of your infrastructure in a remote location. This can be beneficial for a number of reasons, including:

Keeping unencrypted sensitive information off disk

Easily sharing reusable code modules

Working in a team

Enabling version control and auditing

However, remote state does not allow you to delegate output to other teams. This is because the state file contains the full state of your infrastructure, including all of the resources that have been created. If you want to delegate output to other teams, you will need to use a different mechanism, such as a configuration management tool or a messaging system.

upvoted 5 times

 wh1t4k3r Highly Voted 12 months ago

I would appreciate comments on my thoughts: this would be a case of choose 2 right? A is also not a benefit, right?

upvoted 5 times

 alen995454 Most Recent 5 months, 4 weeks ago

**Selected Answer: B**

Saw this on another test engine,, worded slightly different.

"The Terraform State file does not contain the actual module code"

upvoted 1 times

 uax 8 months, 2 weeks ago

**Selected Answer: B**

It's B. Like another person mentioned, sharing reusable code modules is the module registry's job.

<https://developer.hashicorp.com/terraform/language/state/remote>

upvoted 2 times

 govind\_yagyasaini 8 months, 3 weeks ago

<https://developer.hashicorp.com/terraform/language/state/remote>

upvoted 2 times

 RonZhong 11 months ago

**Selected Answer: B**

Remote state wasn't made for sharing modules because that's module registry's job. :)

upvoted 4 times

 Sima\_01 11 months, 1 week ago

itssss BBBB!!!!!!

upvoted 2 times

 Rajmane 11 months, 1 week ago

**Selected Answer: A**

Why not A

upvoted 2 times

 AWS\_cert2023 1 year ago

The answer B.

<https://developer.hashicorp.com/terraform/language/state/remote>

With remote state, Terraform writes the state data to a remote data store, which can then be shared between all members of a team.

Remote state allows you to share output values with other configurations.

For fully-featured remote backends, Terraform can also use state locking to prevent concurrent runs of Terraform against the same state.

upvoted 4 times

 ShakDaddy 11 months, 2 weeks ago

The question states "which is NOT a benefit"

upvoted 2 times

When using multiple configurations of the same Terraform provider, what meta-argument must be included in any non-default provider configurations?

- A. depends\_on
- B. alias
- C. id
- D. name

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Sam1101** 11 months, 2 weeks ago

**Selected Answer: B**

To create multiple configurations for a given provider, include multiple provider blocks with the same provider name. For each additional non-default configuration, use the alias meta-argument to provide an extra name segment.

upvoted 1 times

 **DevoteamAnalytix** 1 year ago

**Selected Answer: B**

<https://developer.hashicorp.com/terraform/language/providers/configuration#alias-multiple-provider-configurations>

upvoted 4 times

A developer accidentally launched a VM (virtual machine) outside of the Terraform workflow and ended up with two servers with the same name. They don't know which VM Terraform manages but do have a list of all active VM IDs.

Which of the following methods could you use to discover which instance Terraform manages?

- A. Run terraform taint/code on all the VMs to recreate them
- B. Update the code to include outputs for the ID of all VMs, then run terraform plan to view the outputs
- C. Run terraform state list to find the names of all VMs, then run terraform state show for each of them to find which VM ID Terraform manages
- D. Use terraform refresh/code to find out which IDs are already part of state

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **SureNot** 1 month ago

Multiple answer? - BC.  
B will also work.

upvoted 1 times

 **dankositze** 4 months, 3 weeks ago

**Selected Answer: C**

I vote C

upvoted 1 times

 **DevoteamAnalytix** 1 year ago

**Selected Answer: C**

<https://developer.hashicorp.com/terraform/cli/commands/state/list>

<https://developer.hashicorp.com/terraform/cli/commands/state/show>

upvoted 4 times

Which of the following is not considered a safe way to inject sensitive values into a Terraform Cloud workspace?

- A. Edit the state file directly just before running terraform apply
- B. Set the variable value on the command line with the -var flag
- C. Write the value to a file and specify the file with the -var-file flag

**Correct Answer: A**

*Community vote distribution*

A (75%)

C (25%)

 **VSMu** 12 months ago

**Selected Answer: A**

Choosing A as it is not right way to do it. Even though C is not preferred, I think you can do it as you are supplying it as a parameter during run and the file is local to the operator running the command and hence can delete it after that. Since we are talking about state being stored remotely, it works very much similar to option B.

upvoted 3 times

 **ShakDaddy** 11 months, 2 weeks ago

A is not possible at all. The question is regarding which is “NOT” the safe way. By interpretation meaning possible but not recommended. The answer is suggests C.

upvoted 1 times

 **biprodatta** 1 year ago

**Selected Answer: C**

Putting sensitive information in file and using that file whole terraform run is not a safe way !

upvoted 1 times

 **AWS\_cert2023** 1 year ago

The answer C.

A is wrong for Terraform state, don't recommend to edit state file.

upvoted 3 times

 **DevoteamAnalytix** 1 year ago

I agree, A is nonsense and B is a favored way, so C is the bad one

upvoted 1 times

If you update the version constraint in your Terraform configuration, Terraform will update your lock file the next time you run terraform init.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

B (93%)

7%

tom\_cat Highly Voted 1 year ago

Selected Answer: B

You will get an error. You have to run terraform init -upgrade to change the version in .terraform.lock.hcl file.  
upvoted 9 times

DevteamAnalytix 1 year ago

Correct: "Use the -upgrade option if you want Terraform to ignore the dependency lock file and consider installing newer versions."

<https://developer.hashicorp.com/terraform/cli/commands/init#plugin-installation>

upvoted 2 times

shax910 8 months ago

Wrong. The command you used is to override the lock file's current version, installing the latest version, and the version tracked in the lock will be ignored.  
upvoted 1 times

Newuser95 Most Recent 4 months, 3 weeks ago

Selected Answer: B

If a particular provider already has a selection recorded in the lock file, Terraform will always re-select that version for installation, even if a new version has become available. You can override that behavior by adding the -upgrade option when you run terraform init, in which case Terraform will disregard the existing selections and once again select the newest available version matching the version constraint.

Tested in lab

upvoted 1 times

shax910 8 months ago

Selected Answer: A

Answer is A

If a particular provider already has a selection recorded in the lock file, Terraform will always re-select that version for installation, even if a new version has become available.

You can override that behavior by adding the -upgrade option when you run terraform init, in which case Terraform will disregard the existing selections and once again select the newest available version matching the version constraint.

<https://developer.hashicorp.com/terraform/language/files/dependency-lock#dependency-installation-behavior-~:text=Terraform%20will%20always%20re%2Dselect%20that%20version%20for%20installation%2C%20even%20if%20a%20new%20version%20has%20become%20available>

upvoted 1 times

VSMu 11 months, 3 weeks ago

Selected Answer: B

While I chose B (false), the question is not clear. Reason , it will not automatically update the version with terraform init, but there is an option upgrade that needs to be added (terraform init -upgrade). Not sure if the exam has an implicit assumption that you run `terraform init` with -upgrade option and if so the answer would be true.

upvoted 3 times

March2023 1 year ago

Selected Answer: B

I believe this is False

upvoted 1 times

You must initialize your working directory before running terraform validate.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

A (100%)

✉️  **DevoteamAnalytix** Highly Voted 1 year ago

**Selected Answer: A**

"Validation requires an initialized working directory with any referenced plugins and modules installed. "  
<https://developer.hashicorp.com/terraform/cli/commands/validate>  
upvoted 5 times

✉️  **LemonadeSoftware** Most Recent 8 months ago

A. True

Before running the terraform validate command, it is recommended to initialize your working directory using the terraform init command. The Terraform init command initializes the working directory and downloads the necessary providers and modules specified in your Terraform configuration. After initialization, you can use Terraform validate to check the syntax and validity of your Terraform files.

upvoted 1 times

If you manually destroy infrastructure, what is the best practice reflecting this change in Terraform?

- A. Manually update the state file
- B. Remove the resource definition from your file and run terraform apply -refresh-only
- C. Run terraform import
- D. It will happen automatically

**Correct Answer: B**

*Community vote distribution*

B (100%)

✉️  **DANDA9989** 10 months ago

**Selected Answer: B**

B. Remove the resource definition from your file and run terraform apply -refresh-only  
upvoted 2 times

✉️  **Pete987** 10 months, 3 weeks ago

B. Remove the resource definition from your file and run terraform apply -refresh-only  
C  
upvoted 2 times

You created infrastructure outside of the Terraform workflow that you now want to manage using Terraform. Which command brings the infrastructure into Terraform state?

- A. terraform init
- B. terraform get
- C. terraform refresh
- D. terraform import

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **DevoteamAnalytix** 1 year ago

**Selected Answer: D**

<https://developer.hashicorp.com/terraform/cli/import>

upvoted 3 times

When using Terraform to deploy resources into Azure, which scenarios are true regarding state files? (Choose two.)

- A. When you change a Terraform-managed resource via the Azure Cloud Console, Terraform updates the state file to reflect the change during the next plan or apply
- B. Changing resources via the Azure Cloud Console records the change in the current state file
- C. When you change a resource via the Azure Cloud Console, Terraform records the changes in a new state file
- D. Changing resources via the Azure Cloud Console does not update current state file

**Correct Answer: AD**

*Community vote distribution*

D (73%)

AD (27%)

 **ogerber** 2 months ago

**Selected Answer: AD**

100% 2 answers are required

upvoted 1 times

👤 **dankositze** 4 months, 3 weeks ago

**Selected Answer: D**

Should not be a "choose two" because (D) is the only answer.

upvoted 1 times

👤 **AkaAka4** 2 months, 2 weeks ago

Refer to an earlier question #98. Answer should be A and D. I honestly suggest anyone who is not that knowledgeable (like me) to not be so confident and mislead others. Thank you.

upvoted 1 times

👤 **azurebot123** 1 month, 3 weeks ago

Yes if you refer to an earlier question #19. Answer should be AD only.

LOL I hope you didn't look up Q19.

Okay, reasons I'm not sure about A:

- you change a Terraform-managed resource via the Azure Cloud Console,
- when you run Terraform plan or apply, Terraform will first do a refresh and update the metadata to what you've changed in cloud console. BUT Terraform will subsequently start applying what's in the config file and revert the change you've made via cloud console. Finally it updates the state file with what's in terraform config.

upvoted 1 times

👤 **azurebot123** 1 month, 3 weeks ago

oh look i cleared my own doubt - reason why it's A:

terraform plan or terraform apply without approval is basically a terraform state refresh. State file will be updated with the changes made on cloud console.

upvoted 1 times

👤 **Spandrop** 7 months ago

**Selected Answer: D**

Only D is the correct one, Terraform will never update the state file to reflect changes that occurred in the environment manually (outside Terraform), it will do the opposite, if the your code is not updated accordingly with what was manually changed, Terraform will bring your environment back to what it is in the state file.

upvoted 2 times

👤 **Tronko86** 8 months, 1 week ago

**Selected Answer: D**

Only D

upvoted 3 times

👤 **Jas14** 9 months, 2 weeks ago

AD is the correct answer

upvoted 4 times

👤 **nuridgsn** 1 year ago

**Selected Answer: D**

A. It is not correct, terraform plan doesn't reflect the changes, it only shows the changes to you when you run terraform apply it reflects the change.

B. When you do manual change in resource, Terraform doesn't imply changes automatically

C. Terraform never creates a new state file for the new changes and it uses the existing one unless you don't specify a different backend.

D. Correct. You can only update tfstate file with either terraform apply (suggested) or terraform apply -refresh-only (not suggested though).

Option A has probably typo issue, terraform plan shouldn't be in there.

upvoted 2 times

👤 **DevteamAnalytix** 1 year ago

**Selected Answer: AD**

For me B and C is wrong, so it's A and D

upvoted 2 times

Which statement describes a goal of infrastructure as code?

- A. A pipeline process to test and deliver software
- B. Defining a vendor-agnostic API
- C. Write once, run anywhere
- D. The programmatic configuration of resources

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **ShakDaddy** 11 months, 2 weeks ago

**Selected Answer: D**

Repeat question but answer is D

upvoted 1 times

 **Jlee7** 1 year ago

D This means that the infrastructure is defined in code, and the code is used to create and manage the infrastructure.

upvoted 1 times

 **DevoteamAnalytix** 1 year ago

Same question like #097 and there the majority votes for D (programmatic config of resources)

upvoted 1 times

terraform validate confirms the syntax of Terraform files.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **inkedia3** 11 months, 1 week ago

**Selected Answer: A**

A. True

Correct, terraform validate is used to confirm the syntax and structure of Terraform files. It checks whether your configuration files are correctly written and can be successfully parsed by Terraform. This helps catch simple errors or typos in your configuration before you proceed with other Terraform commands like terraform plan or terraform apply.

upvoted 3 times

Which command adds existing resources into Terraform state?

- A. terraform init
- B. terraform plan
- C. terraform refresh
- D. terraform import
- E. All of these

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **heeloco** 9 months, 4 weeks ago

**Selected Answer: D**

agree Pete987

upvoted 1 times

 **Pete987** 10 months, 3 weeks ago

D: terraform import

upvoted 4 times

It is best practice to store secret data in the same version control repository as your Terraform configuration.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **Pete987**  10 months, 3 weeks ago

B. False Never save secret data in version control

upvoted 5 times

 **heeloco**  9 months, 4 weeks ago

**Selected Answer: B**

agree Pete987

upvoted 1 times

Which of the following commands would you use to access all of the attributes and details of a resource managed by Terraform?

- A. terraform state list 'provider\_type.name'
- B. terraform state show 'provider\_type.name'
- C. terraform get 'provider\_type.name'
- D. terraform state list

**Correct Answer:** B

*Community vote distribution*

B (100%)

 **LemonadeSoftware** 8 months ago

B

This is an example of what I get if I run "terraform state show aws\_instance.web":

```
aws_instance.web:
resource "aws_instance" "web" {
 ami = "ami-0c55b159cbfafe1f0"
 instance_type = "t2.micro"
 subnet_id = "subnet-12345678"
 tags = {
 Name = "web-instance"
 }
}
```

upvoted 1 times

 **cajif66766** 8 months, 1 week ago

terraform state show "aws.instance.example"

upvoted 1 times

 **heeloco** 9 months, 4 weeks ago

**Selected Answer: B**

agree Pete987

upvoted 1 times

 **Pete987** 10 months, 3 weeks ago

B. terraform state show 'provider\_type.name'

upvoted 4 times

terraform validate confirms that your infrastructure matches the Terraform state file.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **heeloco** 9 months, 4 weeks ago

**Selected Answer: B**

agree Pete987

upvoted 2 times

 **Pete987** 10 months, 3 weeks ago

B. False

The statement is not accurate. terraform validate is a command in Terraform that checks whether the configuration files are syntactically and structurally correct. It validates the configuration syntax and structure but does not confirm that your infrastructure matches the Terraform state file.

To confirm that your infrastructure matches the Terraform state file, you would typically use commands like terraform plan to compare the current state of your configuration with the state recorded in the Terraform state file. This helps you understand the differences between the desired configuration and the actual infrastructure that Terraform is managing.

upvoted 3 times

A senior admin accidentally deleted some of your cloud instances. What does Terraform do when you run `terraform apply`?

- A. Build a completely brand new set of infrastructure
- B. Tear down the entire workspace infrastructure and rebuild it
- C. Rebuild only the instances that were deleted
- D. Stop and generate an error message about the missing instances

**Correct Answer:** C

*Community vote distribution*

C (100%)

 **heeloco** 9 months, 4 weeks ago

**Selected Answer: C**

agree Pete987

upvoted 1 times

 **Pete987** 10 months, 3 weeks ago

C. Rebuild only the instances that were deleted

When you run `terraform apply` after some instances have been accidentally deleted by a senior admin, Terraform will detect the differences between your desired configuration (defined in your Terraform files) and the actual state of your cloud infrastructure (as recorded in the `Terraform state` file). It will then take the necessary actions to reconcile the two.

In this case, Terraform will identify that the instances were deleted and no longer exist in the state file. It will then create new instances to match the desired configuration, effectively rebuilding only the instances that were deleted. This approach is one of the key benefits of using Terraform; it helps maintain the desired infrastructure state even when changes occur outside of Terraform's control.

upvoted 2 times

terraform init creates an example main.tf file in the current directory.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

✉️  **BaburTurk** 10 months, 3 weeks ago

**Selected Answer: B**

The terraform init command does not create an example main.tf file in the current directory. Its primary purpose is to initialize the working directory for Terraform configuration and set up the backend, providers, and modules specified in your configuration. It doesn't generate or modify any configuration files; you need to create your main.tf file yourself to define your infrastructure resources and settings.

upvoted 2 times

✉️  **Pete987** 10 months, 3 weeks ago

b. False

upvoted 1 times

Which argument helps prevent unexpected updates when calling Terraform Registry modules?

- A. count
- B. source
- C. version
- D. lifecycle

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **heeloco** 9 months, 4 weeks ago

**Selected Answer: C**

agree Pete987

upvoted 2 times

 **Pete987** 10 months, 3 weeks ago

C. version

The version argument helps prevent unexpected updates when calling Terraform Registry modules. It allows you to specify the exact version of the module you want to use. This ensures that your configuration consistently uses the same version of the module until you explicitly decide to update it to a newer version.

Using the version argument is a good practice to ensure that your infrastructure remains stable and predictable, as it prevents unintentional changes caused by automatic module updates.

upvoted 3 times

 **bigmic** 11 months ago

version is correct

upvoted 2 times

Setting the TF\_LOG environment variable to DEBUG causes debug messages to be logged into stdout.

- A. True
- B. False

**Correct Answer: A**

*Community vote distribution*

B (60%)

A (40%)

 **MohameDaft** 4 months, 1 week ago

**Selected Answer: A**

Setting the TF\_LOG environment variable to DEBUG causes debug messages to be logged to standard output (stdout) or standard error (stderr)  
upvoted 4 times

 **uax** 8 months, 2 weeks ago

**Selected Answer: B**

"Terraform has detailed logs that you can enable by setting the TF\_LOG environment variable to any value. Enabling this setting causes detail logs to appear on stderr."  
<https://developer.hashicorp.com/terraform/internals/debugging>  
upvoted 1 times

 **meallhour** 10 months, 4 weeks ago

**Selected Answer: B**

B Because it's stdrr  
upvoted 3 times

 **akm\_1010** 11 months ago

**Selected Answer: B**

<https://developer.hashicorp.com/terraform/internals/debugging>  
upvoted 1 times

 **Rajmane** 11 months, 1 week ago

**Selected Answer: B**

B Because it's stdrr  
upvoted 1 times

 **suchar** 11 months, 1 week ago

Correct is B. False - not "stdut" but stderr."

<https://developer.hashicorp.com/terraform/internals/debugging>

"Terraform has detailed logs that you can enable by setting the TF\_LOG environment variable to any value. Enabling this setting causes detail logs to appear on stderr."

upvoted 2 times

How would you output returned values from a child module in the Terraform CLI output?

- A. Declare the output in the root configuration
- B. Declare the output in the child module
- C. Declare the output in both the root and child module
- D. None of the above

**Correct Answer: C**

*Community vote distribution*

C (68%)

B (32%)

✉  **nicecurls** Highly Voted 10 months, 2 weeks ago

**Selected Answer: C**

Answer C

The question asks how to output the value in the cli, if we declare output only in a child module it will not be shown in the cli, we need one mc output in the root configuration

upvoted 5 times

✉  **shax910** 8 months ago

100 % you are correct

upvoted 1 times

✉  **bella** Most Recent 1 month, 2 weeks ago

this question is a replica of question 232. Answer given was B. Declare the output in the child module.

upvoted 1 times

✉  **Newuser95** 4 months, 3 weeks ago

**Selected Answer: B**

In a parent module, outputs of child modules are available in expressions as module.<MODULE NAME>.<OUTPUT NAME>. For example, if a child module named web\_server declared an output named instance\_ip\_addr, you could access that value as module.web\_server.instance\_ip\_addr

upvoted 1 times

✉  **alen995454** 5 months, 4 weeks ago

**Selected Answer: C**

Looked this up on another testing engine ( O'Reilly ) and it has "C" as the correct answer.

upvoted 2 times

✉  **kareem\_ashraf** 7 months, 3 weeks ago

question is weird.. If both in C become any it would be C

upvoted 1 times

✉  **Tronko86** 8 months, 1 week ago

**Selected Answer: C**

C is correct. child>parent and parent>CLI

upvoted 2 times

✉  **3cc17f1** 9 months ago

**Selected Answer: C**

C is correct. To print to the CLI it will also need to be declared in the root config

upvoted 1 times

✉  **Tire** 9 months, 1 week ago

**Selected Answer: B**

There is no need to set the output in the root module.

upvoted 1 times

✉  **enc\_0343** 9 months, 2 weeks ago

**Selected Answer: C**

You can use outputs.tf in the child module and the output block in the root module.

upvoted 1 times

✉  **BaburTurk** 10 months, 3 weeks ago

B. Declare the output in the child module.

To output returned values from a child module in the Terraform CLI output, you need to declare the output in the child module.

Option A, declaring the output in the root configuration, is incorrect because the output is defined in the child module. Option C, declaring the output in both the root and child module, is also incorrect because this would create duplicate outputs. Option D, none of the above, is incorrect because there is a way to output returned values from a child module in the Terraform CLI output.

upvoted 3 times

✉  **Misiek** 10 months, 1 week ago

A child module can use outputs to expose a subset of its resource attributes to a parent module. A root module can use outputs to print certain values in the CLI output after running terraform apply .

upvoted 2 times

✉  **RonZhong** 11 months ago

**Selected Answer: C**

1. To pass a value from child module to parent module, it must declare the output variable in child module.
2. To print out a value, it must declare the output variable in parent module.

upvoted 3 times

✉  **akm\_1010** 11 months ago

**Selected Answer: C**

to output returned values from a child module to CLI - Declare the output in both the root and child module. For example:

child-module.tf

```
output "child_foo" {
 value = "foobar"
}
```

main.tf

```
module "child" {
 source = "path/to/child"
}
```

```
output "output_to_cli" {
 value = module.child.child_foo
}
```

upvoted 3 times

✉  **Rajmane** 11 months, 1 week ago

**Selected Answer: B**

B is correct answer

upvoted 3 times

✉  **ShakDaddy** 11 months, 2 weeks ago

**Selected Answer: B**

In a parent module, outputs of child modules are available in expressions as module.<MODULE NAME>.<OUTPUT NAME>. For example, if a child module named web\_server declared an output named instance\_ip\_addr, you could access that value as module.web\_server.instance\_ip\_addr.

upvoted 3 times

What is the Terraform resource name of the following resource block?

```
resource "azurerm_resource_group" "dev" {
 name = "test"
 location = "westus"
}
```

- A. azurerm\_resource\_group
- B. azurerm
- C. test
- D. dev

**Correct Answer: A**

*Community vote distribution*

D (100%)

 **Bolgarwow** 3 months, 2 weeks ago

49 question ?

upvoted 2 times

 **Roomeer** 5 months, 1 week ago

**Selected Answer: D**

D.dev is the correct answer

upvoted 2 times

 **gold4otas** 6 months, 2 weeks ago

**Selected Answer: D**

"dev" is the resource name or alias., so the answer is "D"

"azurerm\_resource\_group" is the resource type or kind.

upvoted 2 times

 **dev\_maftuna** 7 months, 1 week ago

D.dev is the correct answer

upvoted 4 times

 **vindi135** 7 months, 3 weeks ago

**Selected Answer: D**

D. dev is the name label and the suffix you'd use within terraform to reference this resource.

upvoted 3 times

When do you need to explicitly execute terraform refresh-only?

- A. Before every terraform plan
- B. Before every terraform apply
- C. Before every terraform import
- D. None of the above

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **vindi135** 7 months, 3 weeks ago

**Selected Answer: D**

D. none of the above. The question isn't fully correct, the command is terraform apply -refresh-only. You'd only want to do this in place of terraform taint, when you'd want the state to be the same as the real world infrastructure

upvoted 4 times

 **Alandt** 5 months, 3 weeks ago

init mate? (London accent)

upvoted 2 times

How is the Terraform cloud integration differ from other state backends such as S3, Consul, etc.?

- A. It can execute Terraform runs on dedicated infrastructure in Terraform Cloud
- B. It doesn't show the output of a terraform apply locally
- C. It is only available to paying customers
- D. All of the above

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: A**

A. Terraform cloud runs plan and apply in its own infra  
B. Terraform cloud has nothing to do with outputs. all outputs are displayed by default  
C. Terraform cloud free tier is available.  
D. False

upvoted 2 times

Which of the following are advantages of using infrastructure as code (IaC) instead of provisioning with a graphical user interface (GUI)? (Choose two.)

- A. Secures your credentials
- B. Lets you version, reuse, and share infrastructure configuration
- C. Provisions the same resources at a lower cost
- D. Reduces risk of operator error
- E. Prevents manual modifications to your resources

**Correct Answer:** BE

*Community vote distribution*

BD (71%)

14%

14%

 **dankositze** 4 months, 3 weeks ago

**Selected Answer: BD**

(D) yes but (B) is a bad additional answer.

Hope this one has been deprecated from the actual exam by now but BD seems to be the least worst answer

upvoted 1 times

 **Alandt** 5 months, 2 weeks ago

**Selected Answer: BD**

B. Infrastructure as Code (IaC) allows you to define and manage your infrastructure using code, which can be versioned, reused, and shared. brings benefits such as easier collaboration, tracking changes over time, and ensuring consistency in your infrastructure.

D. Using IaC reduces the risk of operator error compared to provisioning through a graphical user interface (GUI). With IaC, you have a repeat and automated process, which minimizes the chances of human errors that might occur when manually configuring resources through a GUI.

upvoted 1 times

 **gold4otas** 6 months, 2 weeks ago

**Selected Answer: BD**

B. Infrastructure as Code (IaC) allows you to define and manage your infrastructure using code, which can be versioned, reused, and shared. brings benefits such as easier collaboration, tracking changes over time, and ensuring consistency in your infrastructure.

D. Using IaC reduces the risk of operator error compared to provisioning through a graphical user interface (GUI). With IaC, you have a repeat and automated process, which minimizes the chances of human errors that might occur when manually configuring resources through a GUI.

upvoted 1 times

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: BD**

B & D.

upvoted 1 times

 **[Removed]** 7 months, 3 weeks ago

I choose B and D.

upvoted 1 times

 **kareem\_ashraf** 7 months, 3 weeks ago

**Selected Answer: BD**

B-D

Option A is incorrect because IaC does not secure your credentials by default. You will need to use additional security measures, such as encryption, to protect your credentials.

Option C is incorrect because the cost of provisioning resources is typically determined by the cloud provider, and not by the method used to provision the resources.

Option E is incorrect because IaC does not prevent manual modifications to your resources. If a user has access to your IaC files, they can make changes to your infrastructure configuration.

upvoted 1 times

 **kareem\_ashraf** 7 months, 3 weeks ago

**Selected Answer: AD**

A and D

Option A is incorrect because IaC does not secure your credentials by default. You will need to use additional security measures, such as encryption, to protect your credentials.

Option C is incorrect because the cost of provisioning resources is typically determined by the cloud provider, and not by the method used to provision the resources.

Option E is incorrect because IaC does not prevent manual modifications to your resources. If a user has access to your IaC files, they can make changes to your infrastructure configuration.

upvoted 1 times

 **kareem\_ashraf** 7 months, 3 weeks ago

B. And D

upvoted 1 times

 **petersoliman** 7 months, 3 weeks ago

**Selected Answer: AB**

A and B

upvoted 1 times

One cloud configuration always maps to a single remote workspace.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **meetplanet** 4 weeks ago

Answer is A. You should ask chat gpt again, something like "Are you sure?". Then it will say correct answer is A. A cloud configuration is typically designed to map to a specific remote workspace, as it includes unique settings and parameters that define that particular environment.

upvoted 1 times

 **umavaja** 7 months ago

from chatgpt  
False.

In Terraform, a remote workspace is a part of the Terraform Cloud or Terraform Enterprise service that represents a separate instance of your infrastructure. While it's common to have a one-to-one mapping between configurations and workspaces, it's not a strict requirement.

In practice, you can have multiple remote workspaces associated with a single Terraform configuration. Each workspace can have its own state variables, and environment. This allows you to manage different instances or environments of your infrastructure using the same configuration. For example, you might have separate workspaces for development, staging, and production environments, all using the same Terraform configuration.

To summarize, while it's typical to have a one-to-one mapping between a Terraform configuration and a remote workspace, it's not a strict limitation, and you can use multiple workspaces with a single configuration if it suits your organizational or project needs.

upvoted 2 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: B**

B- False

upvoted 1 times

Multiple team members are collaborating on infrastructure using Terraform and want to format their Terraform code following standard Terraform-style convention.

How could they automatically ensure the code satisfies conventions?

- A. Replace all tabs with spaces
- B. Terraform automatically formats configuration on terraform apply
- C. Run terraform validate prior to executing terraform plan or terraform apply
- D. Use terraform fmt

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: D**

use the fmt command

upvoted 2 times

Which backend does the Terraform CLI use by default?

- A. Depends on the cloud provider configured
- B. Remote
- C. Terraform Cloud
- D. Local
- E. HTTP

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: D**

By default, Terraform uses a backend called local , which stores state as a local file on disk.

upvoted 3 times

The Terraform CLI will print output values from a child module after running terraform apply.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (83%)

A (17%)

✉️ **PolitoMex** 7 months, 2 weeks ago

**Selected Answer: B**

child modules send outputs to root module and then the root module prints the outputs.

upvoted 1 times

✉️ **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: A**

True. Output values are stored in state file. Only after a terraform apply, it knows that there are values in state file. Running terraform output output\_name results in warning

Warning: No outputs found.

The state file either has no outputs defined, or all the defined outputs are empty. Please define an output in your configuration with the `output` keyword and run `terraform refresh` for it to become available. If you are using interpolation, please verify the interpolated value is not empty. You can use the `terraform console` command to assist.

upvoted 1 times

✉️ **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: B**

Output values have several uses:

A child module can use outputs to expose a subset of its resource attributes to a parent module.

A root module can use outputs to print certain values in the CLI output after running terraform apply.

upvoted 4 times

What does terraform refresh-only modify?

- A. Your cloud infrastructure
- B. Your Terraform plan
- C. Your Terraform configuration
- D. Your state file

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: D**

refresh-only tries to match state file with real world infra. It does not refer to configuration in tf files.

upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: D**

The terraform refresh command is used to reconcile the state Terraform knows about (via its state file) with the real-world infrastructure. This can be used to detect any drift from the last-known state and to update the state file. This does not modify infrastructure, but does modify the state file

upvoted 1 times

What does terraform import do?

- A. Imports existing resources into the state file
- B. Imports all infrastructure from a given cloud provider
- C. Imports a new Terraform module
- D. Imports clean copies of tainted resources
- E. None of the above

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **gold4otas** 6 months, 2 weeks ago

**Selected Answer: A**

- A. Imports existing resources into the state file  
upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: A**

- lets you target a resource that already exists, and map it to a resource you've defined in code  
upvoted 2 times

Which of the following is the correct way to pass the value in the variable num\_servers into a module with the input server?

- A. servers = var(num\_servers)
- B. \${var.num\_servers}
- C. servers = num\_servers
- D. servers = var.num\_servers

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **boapaulo** 5 months, 1 week ago

same question as question 31  
upvoted 1 times

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: D**

- D. servers = var.num\_servers (HCL2 syntax)  
upvoted 1 times

A developer on your team is going to tear down an existing deployment managed by Terraform and deploy a new one. However, there is a server resource named `aws_instance.ubuntu[1]` they would like to keep. What command should they use to tell Terraform to stop managing that specific resource?

- A. `terraform destroy aws_instance.ubuntu[]`
- B. `terraform apply rm aws_instance.ubuntu[]`
- C. `terraform state rm aws_instance.ubuntu[]`
- D. `terraform plan rm aws_instance.ubuntu[]`

**Correct Answer:** C

*Community vote distribution*

C (100%)

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer:** C

The `'terraform state rm'` command removes bindings from the Terraform state, causing Terraform to "forget about" existing objects.  
upvoted 2 times

Before you can use a remote backend, you must first execute `terraform init`.

- A. True
- B. False

**Correct Answer:** A

*Community vote distribution*

A (100%)

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer:** A

This command initializes a working directory containing Terraform configuration files, downloads provider plugins and sets up the backend for state storage  
upvoted 2 times

What does running a terraform plan do?

- A. Compares your Terraform code and local state file to the remote state file in a cloud provider and determines if any changes need to be made
- B. Imports all of your existing cloud provider resources to the state file
- C. Installs all providers and modules referenced by configuration
- D. Compares the state file to your Terraform code and determines if any changes need to be made

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: D**

it compares your current state (the existing resources) with your desired state (the configuration file) and generates a plan of action to achieve The plan shows you which resources will be created, modified, or destroyed  
upvoted 3 times

Which of the following statements about Terraform modules is not true?

- A. Modules must be publicly accessible
- B. You can call the same module multiple times
- C. A module is a container for one or more resources
- D. Modules can call other modules

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: A**

A. Module sources can be public or private  
upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: A**

A - Modules do not have to be publicly called  
upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

sorry accessible \*

upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

If you're using a private registry, you may ignore this requirement  
upvoted 1 times

How can a ticket-based system slow down infrastructure provisioning and limit the ability to scale? (Choose two.)

- A. End-users have to request infrastructure changes
- B. Ticket based systems generate a full audit trail of the request and fulfillment process
- C. Users can access a catalog of approved resources from drop down lists in a request form
- D. The more resources your organization needs, the more tickets your infrastructure team has to process

**Correct Answer:** AD

*Community vote distribution*

AD (100%)

 **Alandt** 5 months, 3 weeks ago

**Selected Answer:** AD

this is the second time I'm seeing this question

upvoted 3 times

 **AkaAka4** 2 months, 2 weeks ago

This is true for several 200+ questions. It's strange that ExamTopics is not checking for duplicate questions, but it's a good chance to study the questions again.

upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer:** AD

A-D correct

upvoted 1 times

How do you specify a module's version when publishing it to the public Terraform Module Registry?

- A. Configure it in the module's Terraform code
- B. Mention it on the module's configuration page on the Terraform Module Registry
- C. The Terraform Module Registry does not support versioning modules
- D. Tag a release in the associated repo

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: D**

Tag releases using semantic versioning protocol  
upvoted 2 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: D**

release tag names  
upvoted 1 times

What Terraform command always causes a state file to be updated with changes that might have been made outside of Terraform?

- A. terraform plan -refresh-only
- B. terraform show -json
- C. terraform apply -lock=false
- D. terraform plan -target-state

**Correct Answer: D**

*Community vote distribution*

A (100%)

 [Removed] 6 months, 2 weeks ago

**Selected Answer: A**

A - terraform plan -refresh-only

upvoted 3 times

 dev\_maftuna 7 months, 1 week ago

A. terraform plan -refresh-only

upvoted 1 times

 dev\_maftuna 7 months, 2 weeks ago

A. The -refresh-only flag with the terraform plan command updates the state file with the current state of the real resources, without applying any changes. This command compares the current state of the infrastructure with the Terraform configuration and updates the state file with any changes that were made externally.

upvoted 2 times

 Stargazer11 7 months, 3 weeks ago

Neither one of them is correct. Should be terraform apply -refresh-only

upvoted 4 times

 kareem\_ashraf 7 months, 3 weeks ago

non is correct , some thing maybe wrong with this question , should be terraform refresh i think

upvoted 1 times

 vindi135 7 months, 3 weeks ago

**Selected Answer: A**

A. is the closest answer but not quite the right command that would make the changes to the state file. Its terraform apply -refresh-only

upvoted 2 times

 petersoliman 7 months, 3 weeks ago

**Selected Answer: A**

A is the correct answer

upvoted 1 times

Which command must you first run before performing further Terraform operations in a working directory?

- A. terraform plan
- B. terraform workspace
- C. terraform init
- D. terraform import

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **kabary** 7 months ago

**Selected Answer: C**

Answer is C. Easiest question in the whole series lol  
upvoted 3 times

Which command lets you experiment with Terraform expressions?

- A. terraform console
- B. terraform validate
- C. terraform env
- D. terraform test

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **boapaulo** 5 months, 1 week ago  
same question as Question #182  
upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: A**

You can experiment with the behavior of Terraform's expressions from the Terraform expression console  
upvoted 2 times

What kind of configuration block will create an infrastructure object with settings specified within the block?

- A. provider
- B. state
- C. data
- D. resource

**Correct Answer:** D

*Community vote distribution*

D (100%)

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: D**

resource block is used to provision the resource

upvoted 2 times

When do changes invoked by terraform apply take effect?

- A. After Terraform has updated the state file
- B. Once the resource provider has fulfilled the request
- C. Immediately
- D. None of the above are correct

**Correct Answer:** D

*Community vote distribution*

B (100%)

✉  **alen995454** 5 months, 3 weeks ago

**Selected Answer: B**

B happens before A

upvoted 1 times

✉  **petersoliman** 7 months, 2 weeks ago

**Selected Answer: B**

it's supposed to be after the request by the provider is over right?

upvoted 1 times

✉  **vindi135** 7 months, 3 weeks ago

**Selected Answer: B**

terraform waits for the provider to complete the infrastructure change before updating the state

upvoted 2 times

✉  **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: B**

changes invoked by terraform apply take effect once the provider has complete the request - Answer B

upvoted 2 times

✉  **kareem\_ashraf** 7 months, 3 weeks ago

**Selected Answer: B**

Answer Is B

upvoted 2 times

✉  **petersoliman** 7 months, 3 weeks ago

**Selected Answer: B**

B is the correct answer

upvoted 2 times

What is the workflow for deploying new infrastructure with Terraform?

- A. Write Terraform configuration, run terraform init to initialize the working directory or workspace, and run terraform apply
- B. Write Terraform configuration, run terraform show to view proposed changes, and terraform apply to create new infrastructure
- C. Write Terraform configuration, run terraform apply to create infrastructure, use terraform validate to confirm Terraform deployed resources correctly
- D. Write Terraform configuration, run terraform plan to initialize the working directory or workspace, and terraform apply to create the infrastructure

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **Coursesh1** 7 months, 2 weeks ago

terraform init

choose (A)

upvoted 1 times

 **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: A**

After writing the config you need to run terraform init so it can download plugins etc. Terraform plan is optional. Hence A is the only correct answer.

upvoted 1 times

Which of these are features of Terraform Cloud? (Choose two.)

- A. Remote state storage
- B. A web-based user interface (UI)
- C. Automatic backups
- D. Automated infrastructure deployment visualization

**Correct Answer:** AB

*Community vote distribution*

AB (100%)

✉️  **kabary** 7 months ago

**Selected Answer: AB**

I agree Answer is A & B.

upvoted 3 times

✉️  **Ramdi1** 7 months, 3 weeks ago

**Selected Answer: AB**

I think A and B are correct

upvoted 1 times

Which option can not keep secrets out of Terraform configuration files?

- A. A shared credential file
- B. Mark the variable as sensitive
- C. Environment Variables
- D. A -var flag

**Correct Answer: D**

*Community vote distribution*

B (100%)

 **petersoliman** 7 months, 2 weeks ago

**Selected Answer: B**

B is the right answer

upvoted 3 times

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: B**

Question is about whether the sensitive values will be in terraform config files NOT about the state file. so B

upvoted 2 times

 **vindi135** 7 months, 3 weeks ago

**Selected Answer: B**

Marking a variable as sensitive doesn't encourage it to be kept out of the terraform files. It will only be omitted in stout or logs when the terraform command is run

upvoted 3 times

Which of the following is not true of Terraform providers?

- A. An individual person can write a Terraform Provider
- B. A community of users can maintain a provider
- C. HashiCorp maintains some providers
- D. Cloud providers and infrastructure vendors can write, maintain, or collaborate on Terraform providers
- E. None of the above

**Correct Answer: E**

*Community vote distribution*

E (75%)

D (25%)

 **boapaulo** 5 months, 1 week ago

same question as question #8

upvoted 3 times

 **alen995454** 5 months, 3 weeks ago

**Selected Answer: E**

E,, they can most certainly collaborate

upvoted 1 times

 **kabary** 7 months ago

**Selected Answer: E**

Answer is E. None of the above

upvoted 1 times

 **Newuser95** 4 months, 3 weeks ago

Hi kabary, could you please input comments on questions 312 to 316 too please? thanking you for your valuable input

upvoted 1 times

 **mohamedessam4444** 7 months ago

E is the right answer, check this <https://www.examtopics.com/discussions/hashicorp/view/74084-exam-terraform-associate-topic-1-question-discussion/>

upvoted 1 times

 **petersoliman** 7 months, 2 weeks ago

**Selected Answer: D**

D is the right answer, Vendors can't collaborate in one provider

upvoted 1 times

 **PolitoMex** 7 months, 2 weeks ago

Is the D

upvoted 1 times

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: E**

E. None of the above

upvoted 1 times

 **Newuser95** 4 months, 3 weeks ago

Hi Stargazer11, could you please input comments on questions 312 to 316 too please? thanking you for your valuable input

upvoted 1 times

Which Terraform command checks that your configuration syntax is correct?

- A. terraform fmt
- B. terraform validate
- C. terraform init
- D. terraform show

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **kabary** 7 months ago

**Selected Answer: B**

Answer is B. terraform validate

upvoted 2 times

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: B**

terraform validate

upvoted 1 times

terraform validate uses provider APIs to verify your infrastructure settings.

A. True

B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **kabary** 7 months ago

**Selected Answer: B**

Answer is obviously false

upvoted 1 times

 **Stargazer11** 7 months, 3 weeks ago

**Selected Answer: B**

B Validate command doesn't talk to providers

upvoted 1 times

 **vindi135** 7 months, 3 weeks ago

**Selected Answer: B**

The terraform validate command validates the configuration files in a directory, referring only to the configuration and NOT accessing any rem services such as remote state, provider APIs, etc.

upvoted 2 times

You add a new provider to your configuration and immediately run terraform apply in the CLI using the local backend. Why does the apply fail?

A. Terraform needs you to format your code according to best practices first

B. Terraform requires you to manually run terraform plan first

C. The Terraform CLI needs you to log into Terraform Cloud first

D. Terraform needs to install the necessary plugins first

**Correct Answer: D**

*Community vote distribution*

D (100%)

 **J\_boy** 4 months, 3 weeks ago

**Selected Answer: D**

Answer is D

upvoted 1 times

Which of these statements about Terraform Cloud workspaces is false?

- A. They can securely store cloud credentials
- B. They have role-based access controls
- C. You must use the CLI to switch between workspaces
- D. Plans and applies can be triggered via version control system integrations

**Correct Answer:** C

*Community vote distribution*

C (100%)

 **kyochan** 2 months, 3 weeks ago

C. You must use the CLI to switch between workspaces

In Terraform Cloud, you don't necessarily need to use the CLI to switch between workspaces. Instead, you can manage workspaces through Terraform Cloud web interface or programmatically using the Terraform Cloud API.

upvoted 1 times

 **dankositze** 4 months, 3 weeks ago

**Selected Answer: C**

C for sure

upvoted 1 times

What value does the Terraform Cloud private registry provide over the public Terraform Module Registry?

- A. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations
- B. The ability to share modules publicly with any user of Terraform
- C. The ability to tag modules by version or release
- D. The ability to share modules with public Terraform users and members of Terraform Cloud Organizations

**Correct Answer:** A

*Community vote distribution*

A (100%)

 **ogerber** 2 months ago

**Selected Answer: A**

A. The ability to restrict modules to members of Terraform Cloud or Enterprise organizations

Correct: This is a key value of the Terraform Cloud private registry. It allows organizations to control access to their modules, ensuring that only authorized members within the organization can use them. This is particularly important for enterprises that need to manage permissions and maintain security and compliance standards.

upvoted 1 times

Terraform providers are part of the Terraform core binary.

- A. True
- B. False

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **ikedia3** 3 months, 3 weeks ago

**Selected Answer: B**

Terraform is logically split into two main parts: Terraform Core and Terraform Plugins. Terraform Core uses remote procedure calls (RPC) to communicate with Terraform Plugins, and offers multiple ways to discover and load plugins to use. Terraform Plugins expose an implementation for a specific service, such as AWS, or provisioner, such as bash.

upvoted 1 times

Which of the following is not a benefit of adopting infrastructure as code?

- A. Reusability of code
- B. Automation
- C. Graphical User Interface
- D. Versioning

**Correct Answer: C**

*Community vote distribution*

C (100%)

 **ikedia3** 3 months, 3 weeks ago

**Selected Answer: C**

GUI is not one of the benefits of using IaC

upvoted 1 times

Where does the Terraform local backend store its state?

- A. In the terraform.tfstate file
- B. In the .terraform directory
- C. In the terraform.tfstate directory
- D. In the .terraform.lock.hcl file

**Correct Answer: A**

*Community vote distribution*

A (100%)

 **ikedia3** 3 months, 3 weeks ago

**Selected Answer: A**

Stores the state in the terraform.tfstate file

upvoted 2 times

Which of these is true about Terraform's plugin-based architecture?

- A. Terraform can only source providers from the internet
- B. You can create a provider for your API if none exists
- C. Every provider in a configuration has its own state file for its resources
- D. All providers are part of the Terraform core binary

**Correct Answer: B**

*Community vote distribution*

B (100%)

 **eternalamit5** 4 months, 2 weeks ago

Correct ans: B

Terraform's plugin-based architecture allows users to create custom providers to manage resources that aren't supported by default Terraform providers.

upvoted 2 times

 **dankositze** 4 months, 3 weeks ago

**Selected Answer: B**

(B) is correct answer

upvoted 1 times

Your risk management organization requires that new AWS S3 buckets must be private and encrypted at rest. How can Terraform Cloud automatically and proactively enforce this security control?

- A. Auditing cloud storage buckets with a vulnerability scanning tool
- B. With a Sentinel policy, which runs before every apply
- C. With an S3 module with proper settings for buckets
- D. By adding variables to each Terraform Cloud workspace to ensure these settings are always enabled

**Correct Answer:** B

 **Bolgarwow** 3 months, 2 weeks ago

104 question looks like it  
upvoted 1 times

 **d68b7a1** 4 months, 3 weeks ago

The answer is B  
upvoted 1 times

If you don't use the local backend, where does Terraform save resource state?

- A. In the remote backend or Terraform Cloud
- B. On the disk
- C. In memory
- D. In an environment variable

**Correct Answer:** A

You are writing a child Terraform module that provisions an AWS instance. You want to reference the IP address returned by the child module in the root configuration. You name the instance resource "main".

Which of these is the correct way to define the output value?

- ```
output "instance_ip_addr" {  
A.   value = aws_instance.main.private_ip  
}  
  
output "aws_instance.instance_ip_addr" {  
B.   value = ${main.private_ip}  
}  
  
output "instance_ip_addr" {  
C.   return aws_instance.main.private_ip  
}  
  
output "aws_instance.instance_ip_addr" {  
D.   return aws_instance.main.private_ip  
}
```

Correct Answer: A

Community vote distribution

A (100%)

 **akbiyik** 4 months, 2 weeks ago

Selected Answer: A

Each output value exported by a module must be declared using an output block:

```
output "instance_ip_addr" {  
value = aws_instance.server.private_ip  
}
```

The value argument takes an expression whose result is to be returned to the user. In this example, the expression refers to the private_ip attribute exposed by an aws_instance resource defined elsewhere in this module (not shown). Any valid expression is allowed as an output va

<https://developer.hashicorp.com/terraform/language/values/outputs>

upvoted 1 times

When does Sentinel enforce policy logic during a Terraform Cloud run?

- A. Before the plan phase
- B. During the plan phase
- C. Before the apply phase
- D. After the apply phase

Correct Answer: C

Community vote distribution

C (100%)

 **aksliveswithaws** 3 months, 3 weeks ago

Selected Answer: C

<https://developer.hashicorp.com/sentinel/docs/terraform#:~:text=the%20sentinel%20integration%20with%20terraform%20runs%20within%20terraform%20enterprise%20after%20a%20terraform%20plan%20and%20before%20a%20terraform%20apply>

upvoted 1 times

What is terraform refresh-only intended to detect?

- A. Empty state files
- B. Corrupt state files
- C. Terraform configuration code changes
- D. State file drift

Correct Answer: D

 **eternalamit5** 4 months, 2 weeks ago

Correct ans D: The terraform refresh-only command is intended to detect and reconcile any drift between the actual state of the infrastructure and the state described in the Terraform state file.

upvoted 2 times

You should run terraform fmt to rewrite all Terraform configurations within the current working directory to conform to Terraform-style conventions.

- A. True
- B. False

Correct Answer: A

Community vote distribution

A (100%)

 **8876ca1** 4 weeks ago

Selected Answer: A

Its True

upvoted 1 times

 **8876ca1** 4 weeks ago

Its recommended but not mandatory

upvoted 1 times

Why would you use the -replace flag for terraform apply?

- A. You want to force Terraform to destroy a resource on the next apply
- B. You want Terraform to ignore a resource on the next apply
- C. You want to force Terraform to destroy and recreate a resource on the next apply
- D. You want Terraform to destroy all the infrastructure in your workspace

Correct Answer: C

Community vote distribution

C (100%)

 **SilentH** 2 months, 1 week ago

Selected Answer: C

C is the answer. The full syntax is "terraform apply -replace=[ADDRESS]"

Instructs Terraform to plan to replace the resource instance with the given address. This is helpful when one or more remote objects have become degraded, and you can use replacement objects with the same configuration to align with immutable infrastructure patterns. You can use -replace with the -destroy option, and it is only available from Terraform v0.15.2 onwards. For earlier versions, use terraform taint to achieve similar result.

<https://developer.hashicorp.com/terraform/cli/commands/plan#planning-options>

upvoted 1 times

You can configure Terraform to log to a file using the TF_LOG environment variable.

- A. True
- B. False

Correct Answer: A

Community vote distribution

B (100%)

 **kyochan** 2 months, 3 weeks ago

Selected Answer: B

You can configure Terraform to log to a file by setting the TF_LOG_PATH environment variable. When this variable is set, Terraform will write its log output to the specified file.

upvoted 1 times

 **6957dbd** 4 months, 1 week ago

This is a badly worded question...

Terraform has detailed logs that you can enable by setting the TF_LOG environment variable to any value. Enabling this setting causes detailed logs to appear on stderr.

To persist logged output you can set TF_LOG_PATH in order to force the log to always be appended to a specific file when logging is enabled. Note that even when TF_LOG_PATH is set, TF_LOG must be set in order for any logging to be enabled.

So, TF_LOG_PATH sets the directory itself, but it will not output anything unless TF_LOG is set.

upvoted 1 times

 **J_boy** 4 months, 3 weeks ago

Selected Answer: B

TF_LOG_PATH is used instead

<https://developer.hashicorp.com/terraform/internals/debugging>

upvoted 3 times

When does Terraform create the .terraform.lock.hcl file?

- A. After your first terraform plan
- B. After your first terraform apply
- C. After your first terraform init
- D. Whenever you enable state locking

Correct Answer: C

 **Colz** Highly Voted 4 months, 1 week ago

Answer C

Terraform automatically creates or updates the dependency lock file each time you run the terraform init command.

upvoted 7 times

 **iamabhi** Most Recent 4 months, 1 week ago

answer D is correct.

upvoted 2 times