

# HashiCorp.TA-002-P.v2021-09-08.q98

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	98
<b>Version:</b>	v2021-09-08
<b># of views:</b>	2951
<b># of Questions views:</b>	78282

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2021-09-08.q98.html>

## NEW QUESTION: 1

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

You can change your backend configuration at any time. You can change both the configuration itself as well as the type of backend (for example from "consul" to "s3"). Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

## NEW QUESTION: 2

What is the result of the following terraform function call?

- A. hello
- B. what?
- C. goodbye

**Answer: ([SHOW ANSWER](#))**

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

## NEW QUESTION: 3

You can migrate the Terraform backend but only if there are no resources currently being managed.

**A.** False

**B.** True

**Answer:** A ([LEAVE A REPLY](#))

Explanation

If you need to migrate to another backend, such as Terraform Cloud, so you can continue managing it. By migrating your Terraform state, you can hand off infrastructure without de-provisioning anything.

<https://www.terraform.io/docs/cloud/migrate/index.html>

#### **NEW QUESTION: 4**

After creating a new workspace "PROD" you need to run the command `terraform select PROD` to switch to it.

**A.** False

**B.** True

**Answer:** A ([LEAVE A REPLY](#))

Explanation

By default, when you create a new workspace you are automatically switched to it To create a new workspace and switch to it, you can use `terraform workspace new <new_workspace_name>`; to switch to a existing workspace you can use `terraform workspace select <existing_workspace_name>`; Example:

\$ `terraform workspace new example`

Created and switched to workspace "example"!

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state for this configuration.

#### **NEW QUESTION: 5**

What is the provider for this fictitious resource?

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

**A.** vpc

**B.** main

**C.** aws

**D.** test

**Answer:** ([SHOW ANSWER](#))

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

#### **NEW QUESTION: 6**

You have multiple developers working on a terraform project (using terraform OSS), and have saved the terraform state in a remote S3 bucket . However ,team is intermittently experiencing inconsistencies in the provisioned infrastructure / failure in the code . You have traced this problem to simultaneous/concurrent runs of terraform apply command for 2/more developers . What can you do to fix this problem?

- A.** Use terraform workspaces feature, this will fix this problem by default , as every developer will have their own state file , and terraform will merge them on server side on its own.
- B.** Structure your team in such a way that only one individual will run terraform apply , everyone will just make changes and share with him. Then there will be no chance of any inconsistencies.
- C.** Stop using remote state , and store the developer tfstate in their own machine . Once a day , all developers should sit together and merge the state files manually , to avoid any inconsistencies.
- D.** Enable terraform state locking for the S3 backend using DynamoDB table. This prevents others from acquiring the lock and potentially corrupting your state.

**Answer:** D ([LEAVE A REPLY](#))

Explanation

S3 backend support state locking using DynamoDB.

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

## NEW QUESTION: 7

After running into issues with Terraform, you need to enable verbose logging to assist with troubleshooting the error. Which of the following values provides the MOST verbose logging?

- A.** ERROR
- B.** INFO
- C.** WARN
- D.** TRACE
- E.** DEBUG

**Answer:** D ([LEAVE A REPLY](#))

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

Examples:

```
export TF_LOG=DEBUG  
export TF_LOG=TRACE
```

## NEW QUESTION: 8

A colleague has informed you that a new version of a Terraform module that your team hosts on an Amazon S3 bucket is broken. The Amazon S3 bucket has versioning enabled. Your colleague tells you to make sure you are not using the latest version in your configuration. You have the following configuration block in your code that refers to the module:

```
module "infranet" { source = "s3::https://s3-us-west-2.amazonaws.com/infrabucket/infra_module.zip"} What is the best way to ensure that you are not using the latest version of the module?
```

- A. Add a module version constraint in your configuration's backend block and specify a previous version.
- B. Add a version key to the module configuration and specify a previous version.
- C. Delete the latest version of the module in S3 to rollback to the previous version.
- D. Add a version property to the module in Terraform's state file and specify a previous version.

**Answer:** ([SHOW ANSWER](#))

Explanation

Version constraints are supported only for modules installed from a module registry, such as the 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.

Only Terraform Registries support module versioning by using the version key, one cannot configure a previous version of the module in the configuration. Deleting the latest version of the module in S3 is the only option of the available options that ensures you won't use the latest version. You could also modify the source URL to specify a versionId URL parameter for a previous version.

<https://www.terraform.io/docs/configuration/modules.html#source>

## NEW QUESTION: 9

Terraform init can indeed be run only a few times, because, every time terraform init will initialize the project

, and download all plugins from the internet repository , regardless of whether they were present or not , and this increases the waiting time

- A. True
- B. False

**Answer:** B ([LEAVE A REPLY](#))

Explanation

Re-running init with modules already installed will install the sources for any modules that were added to configuration since the last init, but will not change any already-installed

modules. Use `-upgrade` to override this behavior, updating all modules to the latest available source code.

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

### **NEW QUESTION: 10**

What is the default backend for Terraform?

- A. consul
- B. gcs
- C. local
- D. etcd

**Answer: C ([LEAVE A REPLY](#))**

Explanation

By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to.

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

### **NEW QUESTION: 11**

Complete the following sentence:

For local state, the workspaces are stored directly in a \_\_\_\_\_.

- A. a file called `terraform.tfstate.backup`
- B. directory called `terraform.workspaces.tfstate`
- C. a file called `terraform.tfstate`
- D. directory called `terraform.tfstate.d`

**Answer: ([SHOW ANSWER](#))**

Explanation

For local state, Terraform stores the workspace states in a directory called `terraform.tfstate.d`.

<https://www.terraform.io/docs/state/workspaces.html#workspace-internals>

### **NEW QUESTION: 12**

Which of the following terraform subcommands could be used to remove the lock on the state for the current configuration?

- A. `Unlock`
- B. `force-unlock`
- C. Removing the lock on a state file is not possible
- D. `state-unlock`

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 13**

Anyone can publish and share modules on the Terraform Public Module Registry, and meeting the requirements for publishing a module is extremely easy. Select from the following list all valid requirements.

(select three)

- A. The module must be PCI/HIPPA compliant.
- B. Module repositories must use this three-part name format, `terraform-- .`
- C. The registry uses tags to identify module versions.
- D. Release tag names must be for the format `x.y.z`, and can optionally be prefixed with a `v`.
- E. The module must be on GitHub and must be a public repo.

**Answer:** C,D,E ([LEAVE A REPLY](#))

Explanation

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

#### NEW QUESTION: 14

Multiple configurations for the same provider can be used in a single configuration file.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it creates an additional provider configuration. For providers that have no required configuration arguments, the implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-provider-instances>

### NEW QUESTION: 15

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address aws\_instance.web refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.
- C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.
- D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer:** ([SHOW ANSWER](#))

Explanation

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:

module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
  # ...  
  count = 4  
}
```

An address like this:

aws\_instance.web[3]

Refers to only the last instance in the config, and an address like this:

aws\_instance.web

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

### NEW QUESTION: 16

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

**Answer:** ([SHOW ANSWER](#))

Reference: <https://github.com/hashicorp/terraform/issues/19291>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

### NEW QUESTION: 17

When TF\_LOG\_PATH is set, TF\_LOG must be set in order for any logging to be enabled.

- A. False

- B. True

**Answer:** B ([LEAVE A REPLY](#))

Explanation

TF\_LOG\_PATH specifies where the log should persist its output to. Note that even when TF\_LOG\_PATH is set, TF\_LOG must be set in order for any logging to be enabled.

For example, to always write the log to the directory you're currently running terraform from:

```
export TF_LOG_PATH=./terraform.log  
export TF_LOG=TRACE
```

### NEW QUESTION: 18

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

- A. True

- B. False

**Answer:** A ([LEAVE A REPLY](#))

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

### NEW QUESTION: 19

Your team lead does not trust the junior terraform engineers who now have access to the git repo . So , he wants you to have some sort of a checking layer , whereby , you can ensure that the juniors will not create any non-compliant resources that might lead to a security audit failure in future. What can you do to efficiently enforce this?

- A.** Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.
- B.** Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them.
- C.** Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code.
- D.** Create a git master branch , and implement PR . Every change needs to be reviewed by you , before being merged to the master branch.

**Answer:** B ([LEAVE A REPLY](#))

Explanation

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### **NEW QUESTION: 20**

True or False? By default, Terraform destroy will prompt for confirmation before proceeding.

- A.** True
- B.** False

**Answer:** A ([LEAVE A REPLY](#))

### **NEW QUESTION: 21**

In regards to deploying resources in multi-cloud environments, what are some of the benefits of using Terraform rather than a provider's native tooling? (select three)

- A.** Terraform can help businesses deploy applications on multiple clouds and on-premises infrastructure.
- B.** Terraform is not cloud-agnostic and can be used to deploy resources across a single public cloud.
- C.** Terraform simplifies management and orchestration, helping operators build large-scale, multi-cloud infrastructure.
- D.** Terraform can manage cross-cloud dependencies.

**Answer:** A,C,D ([LEAVE A REPLY](#))

Explanation

Terraform is cloud-agnostic and allows a single configuration to be used to manage multiple providers, and to even handle cross-cloud dependencies. This simplifies management and orchestration, helping operators build large-scale multi-cloud infrastructures.

<https://www.terraform.io/intro/use-cases.html>

## **NEW QUESTION: 22**

terraform validate validates the syntax of Terraform files.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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

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

## **NEW QUESTION: 23**

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

- 1. max
  - 2. min
  - 3. join
  - 4. replace
  - 5. list
  - 6. length
  - 7. range
- A. Backends
  - B. Data sources
  - C. Named values
  - D. Functions

**Answer: D ([LEAVE A REPLY](#))**

Explanation

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not support user-defined functions, and only the functions built into the language are available for use.

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

## **NEW QUESTION: 24**

In the example below, the depends\_on argument creates what type of dependency?

- A. implicit dependency
- B. non-dependency resource
- C. internal dependency
- D. explicit dependency

**Answer: D ([LEAVE A REPLY](#))**

## **NEW QUESTION: 25**

The terraform init 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.

A. False

B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 26**

If a module uses a local variable, you can expose that value with a terraform output.

A. True

B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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>

### **NEW QUESTION: 27**

Which Terraform command will force a marked resource to be destroyed and recreated on the next apply?

A. terraform fmt

B. terraform destroy

C. terraform taint

D. terraform refresh

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may

need to be modified to reflect the potentially new IP address of a tainted server. The plan command will show this if this is the case.

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

### **NEW QUESTION: 28**

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**

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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>

### **NEW QUESTION: 29**

From the answers below, select the advantages of using Infrastructure as Code.

- A. Provide a codified workflow to develop customer-facing applications.**
- B. Safely test modifications using a "dry run" before applying any actual changes.**
- C. Easily integrate with application workflows (GitLab Actions, Azure DevOps, CI/CD tools).**
- D. Easily change and update existing infrastructure.**
- E. Provide reusable modules for easy sharing and collaboration.**

**Answer: B,C,D,E ([LEAVE A REPLY](#))**

Explanation

Infrastructure as Code is not used to develop applications, but it can be used to help deploy or provision those applications to a public cloud provider or on-premises infrastructure.

All of the others are benefits to using Infrastructure as Code over the traditional way of managing infrastructure, regardless if it's public cloud or on-premises.

### **NEW QUESTION: 30**

Which of the following is an invalid variable name?

- A. count**
- B. web**
- C. var1**
- D. instance\_name**

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/intro/examples/count.html>

### NEW QUESTION: 31

Ric wants to enable detail logging and he wants highest verbosity of logs. Which of the following environment variable settings is correct option for him to select.

- A. Set TF\_LOG = DEBUG
- B. Set VAR\_TF = TRACE
- C. Set TF\_LOG = TRACE
- D. Set VAR\_TF\_LOG = TRACE

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/internals/debugging.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

[\(324 Q&As Dumps, 30%OFF](https://www.fast2test.com/TA-002-P-premium-file.html)

**Special Discount: freecram**)

### NEW QUESTION: 32

Terraform Enterprise (also referred to as pTFE) requires what type of backend database for a clustered deployment?

- A. PostgreSQL
- B. Cassandra
- C. MySQL
- D. MSSQL

**Answer:** ([SHOW ANSWER](#))

Explanation

External Services mode stores the majority of the stateful data used by the instance in an external PostgreSQL database and an external S3-compatible endpoint or Azure blob storage. There is still critical data stored on the instance that must be managed with snapshots. Be sure to check the PostgreSQL Requirements for information that needs to be present for Terraform Enterprise to work. This option is best for users with expertise managing PostgreSQL or users that have access to managed PostgreSQL offerings like AWS RDS.

### NEW QUESTION: 33

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

**Answer: C ([LEAVE A REPLY](#))**

Explanation

terraform workspace list

The command will list all existing workspaces.

Reference: <https://www.terraform.io/docs/cli/commands/workspace/list.html>

#### **NEW QUESTION: 34**

Multiple provider instances blocks for AWS can be part of a single configuration file?

- A. False
- B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it creates an additional provider configuration. For providers that have no required configuration arguments, the implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-provider-instances>

#### **NEW QUESTION: 35**

Your developers are facing a lot of problem while writing complex expressions involving difficult interpolations . They have to run the terraform plan every time and check whether there are errors , and also check terraform apply to print the value as a temporary output for debugging purposes. What should be done to avoid this?

- A.** Use terraform console command to have an interactive UI with full access to the underlying terraform state to run your interpolations , and debug at real-time.
- B.** Add a breakpoint in your code, using the watch keyword , and output the value to console for temporary debugging.
- C.** Use terraform zipmap function , it will be able to easily do the interpolations without complex code.
- D.** Use terraform console command to have an interactive UI , but you can only use it with local state , and it does not work with remote state.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The terraform console command provides an interactive console for evaluating expressions. This is useful for testing interpolations before using them in configurations, and for interacting with any values currently saved in state.

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

### **NEW QUESTION: 36**

Which of the below configuration file formats are supported by Terraform? (Select TWO)

- A.** Node
- B.** JSON
- C.** Go
- D.** YAML
- E.** HCL

**Answer: B,E ([LEAVE A REPLY](#))**

Explanation

Terraform supports both HashiCorp Configuration Language (HCL) and JSON formats for configurations.

<https://www.terraform.io/docs/configuration/>

### **NEW QUESTION: 37**

Which of the below datatype is not supported by Terraform.

- A.** Map
- B.** List
- C.** Object
- D.** Array

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 38**

You have created a custom variable definition file `my_vars.tfvars`. How will you use it for provisioning infrastructure?

- A. `terraform apply -var-state-file ="my_vars.tfvars"`
- B. `terraform apply var-file="my_vars.tfvars"`
- C. `terraform plan -var-file="my_vars.tfvar"`
- D. `terraform apply -var-file="my_vars.tfvars"`

**Answer:** ([SHOW ANSWER](#))

Explanation

To set lots of variables, it is more convenient to specify their values in a variable definitions file (with a filename ending in either `.tfvars` or `.tfvars.json`) and then specify that file on the command line with `-var-file`:

`terraform apply -var-file="my_vars.tfvars"`

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

### NEW QUESTION: 39

Which of the following value will be accepted for `var1`?

```
variable "var1" {  
  type = string  
}
```

- A. None of the above
- B. Both A and B
- C. "5"
- D. 5

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform automatically converts number and bool values to strings when needed.

### NEW QUESTION: 40

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer:** A ([LEAVE A REPLY](#))

Explanation

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

### **NEW QUESTION: 41**

What Terraform feature is shown in the example below?

- A. local values
- B. conditional expression
- C. data source
- D. dynamic block

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 42**

Terraform providers are always installed from the Internet.

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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>

### **NEW QUESTION: 43**

Select the operating systems which are supported for a clustered Terraform Enterprise:  
(select four)

- A. Unix
- B. Red Hat
- C. CentOS
- D. Amazon Linux
- E. Ubuntu

**Answer: B,C,D,E ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/enterprise/before-installing/index.html#operating-systemrequirements>

### **NEW QUESTION: 44**

Dawn has created the below child module. Without changing the module, can she override the instance\_type from t2.micro to t2.large from her code while calling this module?

```
1. resource "aws_instance" "myec2"
2. {
3.   ami = "ami-082b5a644766e0e6f"
4.   instance_type = "t2.micro"
5. }
```

A. YES

B. No

**Answer: B ([LEAVE A REPLY](#))**

Explanation

As the instance\_type is hard-coded in source module, you will not be able to change its value from destination module. Instead of hard-coding you should use variable with default values.

### NEW QUESTION: 45

The following is a snippet from a Terraform configuration file:

Which, when validated, results in the following error:

Fill in the blank in the error message with the correct string from the list below.

A. version

B. multi

C. label

D. alias

**Answer: D ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-providerinstances>

### NEW QUESTION: 46

Talal is a DevOps engineer and he has deployed the production infrastructure using Terraform. He is using a very large configuration file to maintain and update the actual infrastructure. As the infrastructure have grown to a very complex and large, he has started experiencing slowness when he runs terraform plan. What are the options for him to resolve this slowness?

A. Use -refresh=true flag as well as the -target flag with terraform plan in order to work around this.

B. Run terraform refresh every time before running terraform plan.

C. Break large configurations into several smaller configurations that can each be independently applied.

D. Use -refresh=false flag as well as the -target flag with terraform plan in order to work around this.

**Answer: C,D ([LEAVE A REPLY](#))**

## Explanation

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to query multiple resources at once, and the round trip time for each resource is hundreds of milliseconds. On top of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

Although 'Use `-refresh=false` flag as well as the `-target` flag with `terraform plan` in order to work around this.' is a solution, but it's not always recommended. Instead of using `-target` as a means to operate on isolated portions of very large configurations, prefer instead to break large configurations into several smaller configurations that can each be independently applied. Data sources can be used to access information about resources created in other configurations, allowing a complex system architecture to be broken down into more manageable parts that can be updated independently.

Option 'Run `terraform refresh` every time before running `terraform plan`.' and 'Use `-refresh=true` flag as well as the `-target` flag with `terraform plan` in order to work around this.' is not correct because in both the cases `terraform` will query every resources of the infrastructure.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 47

You have created a custom variable definition file `testing.tfvars`. How will you use it for provisioning infrastructure?

- A. `terraform apply -var-state-file ="testing.tfvars"`
- B. `terraform plan -var-file="testing.tfvar"`
- C. `terraform apply -var-file="testing.tfvars"`
- D. `terraform apply var-file="testing.tfvars"`

**Answer: C ([LEAVE A REPLY](#))**

## Explanation

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

## NEW QUESTION: 48

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

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

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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>

### **NEW QUESTION: 49**

You cannot publish your own modules on the Terraform Registry.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Anyone can publish and share modules on the Terraform Registry.

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

### **NEW QUESTION: 50**

A single terraform resource file that defines an aws\_instance resource can simple be renamed to azurerm\_virtual\_machine in order to switch cloud providers

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Providers usually require some configuration of their own to specify endpoint URLs, regions, authentication settings.

Providers Initialization can be done by either explicitly via a provider block or by adding a resource from that provider

<https://www.terraform.io/docs/configuration/providers.html>

### **NEW QUESTION: 51**

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

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 52**

When using parent/child modules to deploy infrastructure, how would you export a value from one module to import into another module.

For example, a module dynamically deploys an application instance or virtual machine, and you need the IP address in another module to configure a related DNS record in order to reach the newly deployed application.

- A. Export the value using terraform export and input the value using terraform input.
- B. Configure the pertinent provider's configuration with a list of possible IP addresses to use.
- C. Configure an output value in the application module in order to use that value for the DNS module.
- D. Preconfigure the IP address as a parameter in the DNS module.

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Output values are like the return values of a Terraform module, and 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.
- \* When using remote state, root module outputs can be accessed by other configurations via a terraform\_remote\_state data source.

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

### **NEW QUESTION: 53**

True or False. The terraform refresh command is used to reconcile the state Terraform knows about (via its state file) with the real-world infrastructure. If drift is detected between the real-world infrastructure and the last known-state, it will modify the infrastructure to correct the drift.

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

Explanation

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

### **NEW QUESTION: 54**

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.

**Answer: C,D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 55**

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

**Answer: A,B,D ([LEAVE A REPLY](#))**

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.>

### **NEW QUESTION: 56**

In order to make a Terraform configuration file dynamic and/or reusable, static values should be converted to use what?

- A. Input Parameters
- B. Module
- C. Regular Expressions
- D. Output Value

**Answer: ([SHOW ANSWER](#))**

Explanation

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module's own source code, and allowing modules to be shared between different configurations.

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

### **NEW QUESTION: 57**

When Terraform needs to be installed in a location where it does not have internet access to download the installer and upgrades, the installation is generally known as to be \_\_\_\_\_.

- A. a private install
- B. disconnected
- C. air-gapped
- D. non-traditional

**Answer: D ([LEAVE A REPLY](#))**

## Explanation

A Terraform Enterprise install that is provisioned on a network that does not have Internet access is generally known as an air-gapped install. These types of installs require you to pull updates, providers, etc. from external sources vs. being able to download them directly.

## NEW QUESTION: 58

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

**Answer: A ([LEAVE A REPLY](#))**

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

## NEW QUESTION: 59

Your organization has moved to AWS and has manually deployed infrastructure using the console. Recently, a decision has been made to standardize on Terraform for all deployments moving forward.

What can you do to ensure that all existing is managed by Terraform moving forward without interruption to existing services?

- A. Submit a ticket to AWS and ask them to export the state of all existing resources and use terraform import to import them into the state file.
- B. Delete the existing resources and recreate them using new a Terraform configuration so Terraform can manage them moving forward.
- C. Resources that are manually deployed in the AWS console cannot be imported by Terraform.
- D. Using terraform import, import the existing infrastructure into your Terraform state.

**Answer: ([SHOW ANSWER](#))**

## Explanation

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

## NEW QUESTION: 60

What are some of the problems of how infrastructure was traditionally managed before Infrastructure as Code?

(select three)

- A. Requests for infrastructure or hardware required a ticket, increasing the time required to deploy applications
- B. Traditional deployment methods are not able to meet the demands of the modern business where resources tend to live days to weeks, rather than months to years
- C. Traditionally managed infrastructure can't keep up with cyclic or elastic applications
- D. Pointing and clicking in a management console is a scalable approach and reduces human error as businesses are moving to a multi-cloud deployment model

**Answer: A,B,C ([LEAVE A REPLY](#))**

Explanation

Businesses are making a transition where traditionally-managed infrastructure can no longer meet the demands of today's businesses. IT organizations are quickly adopting the public cloud, which is predominantly API-driven. To meet customer demands and save costs, application teams are architecting their applications to support a much higher level of elasticity, supporting technology like containers and public cloud resources.

These resources may only live for a matter of hours; therefore the traditional method of raising a ticket to request resources is no longer a viable option. Pointing and clicking in a management console is NOT scale and increases the chance of human error.

## NEW QUESTION: 61

Named workspaces are not a suitable isolation mechanism for strong separation between staging and production?

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

Organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 62

What is the name assigned by Terraform to reference this resource?

```
resource "azurerm_resource_group" "dev" {
    name = "test"
    location = "westus"
}
```

- A. azurerm\_resource\_group
- B. dev
- C. test
- D. azurerm

**Answer:** C ([LEAVE A REPLY](#))

## NEW QUESTION: 63

Which option can not be used to keep secrets out of Terraform configuration files?

- A. A Terraform provider
- B. Environment variables
- C. A -var flag

D. secure string

**Answer: (SHOW ANSWER)**

Reference: <https://secrethub.io/blog/secret-management-for-terraform/>

#### **NEW QUESTION: 64**

When does terraform apply reflect changes in the cloud environment?

- A. Based on the value provided to the -refresh command line argument
- B. None of the above
- C. After updating the state file
- D. Immediately

E. However long it takes the resource provider to fulfill the request

**Answer: B (LEAVE A REPLY)**

#### **NEW QUESTION: 65**

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

**Answer: B (LEAVE A REPLY)**

Explanation

The variable declaration can also include a default argument.

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

#### **NEW QUESTION: 66**

You have provisioned some aws resources in your test environment through Terraform for a POC work. After the POC, now you want to destroy the resources but before destroying them you want to check what resources will be getting destroyed through terraform. what are the options of doing that? (Select TWO)

- A. Use terraform destroy command
- B. This is not possible
- C. Use terraform plan command
- D. Use terraform plan -destroy command.

**Answer: (SHOW ANSWER)**

Explanation

<https://learn.hashicorp.com/terraform/getting-started/destroy>

#### **NEW QUESTION: 67**

The Security Operations team of ABC Enterprise wants to mandate that all the Terraform configuration that creates an S3 bucket must have encryption feature enabled. What is the best way to achieve it?

- A. Use Sentinel Policies.
- B. Use S3 bucket policy.
- C. Create a script that checks the encryption parameter is enabled on every git commit.
- D. Shared a SOP to engineers to mandate encryption feature on S3.

**Answer:** ([SHOW ANSWER](#))

Explanation

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

Using Sentinel with Terraform Cloud involves:

- \* Defining the policies - Policies are defined using the policy language with imports for parsing the Terraform plan, state and configuration.
- \* Managing policies for organizations - Users with permission to manage policies can add policies to their organization by configuring VCS integration or uploading policy sets through the API. They also define which workspaces the policy sets are checked against during runs. (More about permissions.)
- \* 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.
- \* Mocking Sentinel Terraform data - Terraform Cloud provides the ability to generate mock data for any run within a workspace. This data can be used with the Sentinel CLI to test policies before deployment.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

## NEW QUESTION: 68

terraform refresh will update the state file?

- A. True
- B. False

**Answer:** A ([LEAVE A REPLY](#))

Explanation

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. If the state is changed, this may cause changes to occur during the next plan or apply.

## NEW QUESTION: 69

John is writing a module and within the module, there are multiple places where he has to use the same conditional expression but he wants to avoid repeating the same values or

expressions multiple times in a configuration,. What is a better approach to dealing with this?

- A. Local Values
- B. Expressions
- C. Functions
- D. Variables

**Answer: A ([LEAVE A REPLY](#))**

Explanation

A local value assigns a name to an expression, allowing it to be used multiple times within a module without repeating it.

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

### **NEW QUESTION: 70**

Which one is the right way to import a local module names consul?

- A. module "consul" { source = "consul"}
- B. module "consul" { source = "./consul"}
- C. module "consul" { source = "../consul"}
- D. module "consul" { source = "module/consul"}

**Answer: B,C ([LEAVE A REPLY](#))**

Explanation

A local path must begin with either ./ or ../ to indicate that a local path is intended, to distinguish from a module registry address.

```
module "consul" {
  source = "./consul"
}
```

### **NEW QUESTION: 71**

Only the user that generated a plan may apply it.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

The optional -out argument can be used to save the generated plan to a file for later execution with terraform apply, which can be useful when running Terraform in automation.  
Reference: <https://learn.hashicorp.com/tutorials/terraform/automate-terraform>

### **NEW QUESTION: 72**

Which statements best describes what the local variable assignment is doing in the following code snippet:

- A. Create a list of route table names eliminating duplicates
- B. Create a distinct list of route table name objects

C. Create a map of route table names from a list of subnet names

D. Create a map of route table names to subnet names

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 73**

What is the command you can use to set an environment variable named "var1" of type String?

A. export TF\_VAR\_VAR1

B. set TF\_VAR\_var1

C. variable "var1" { type = "string"}

D. export TF\_VAR\_var1

**Answer:** D ([LEAVE A REPLY](#))

Explanation

The environment variable must be in the format TF\_VAR\_name, so for the question:

TF\_VAR\_var1 is the correct choice.

[https://www.terraform.io/docs/commands/environment-variables.html#tf\\_var\\_name](https://www.terraform.io/docs/commands/environment-variables.html#tf_var_name)

### **NEW QUESTION: 74**

A user creates three workspaces from the command line - prod, dev, and test. Which of the following commands will the user run to switch to the dev workspace?

A. terraform workspace dev

B. terraform workspace select dev

C. terraform workspace -switch dev

D. terraform workspace switch dev

**Answer:** B ([LEAVE A REPLY](#))

Explanation

The terraform workspace select command is used to choose a different workspace to use for further operations.

<https://www.terraform.io/docs/commands/workspace/select.html>

### **NEW QUESTION: 75**

Which of the following actions are performed during a terraform init?

A. Initializes downloaded and/or installed providers

B. Initializes the backend configuration

C. Provisions the declared resources in your configuration

D. Download the declared providers which are supported by HashiCorp

**Answer:** A,B,D ([LEAVE A REPLY](#))

Explanation

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

This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times. 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.

terraform init command does -

- \* Copy a Source Module
- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

### **NEW QUESTION: 76**

Select all features which are exclusive to Terraform Enterprise. (Select Three)

- A.** Sentinel
- B.** Cost Estimation
- C.** Audit Logs
- D.** Clustering
- E.** SAML/SSO

**Answer: C,D,E ([LEAVE A REPLY](#))**

Explanation

Sentinel and Cost Estimation are also available in Terraform Cloud

<https://www.hashicorp.com/products/terraform/pricing/>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

### **NEW QUESTION: 77**

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

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

**Answer: D ([LEAVE A REPLY](#))**

## **NEW QUESTION: 78**

Refer to the following terraform variable definition

```
variable "track_tag" { type = list default = ["data_ec2","integration_ec2","digital_ec2"]}  
track_tag = { Name = element(var.track_tag,count.index)} If count.index is set to 2, which  
of the following values will be assigned to the name attribute of track_tag variable?
```

- A. digital\_ec2
- B. track\_tag
- C. integration\_ec2
- D. data\_ec2

**Answer: A ([LEAVE A REPLY](#))**

## **NEW QUESTION: 79**

State locking does not happen automatically and must be specified at run

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

State locking happens automatically on all operations that could write state.

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

## **NEW QUESTION: 80**

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

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

**Answer: D ([LEAVE A REPLY](#))**

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

## **NEW QUESTION: 81**

What does terraform plan do ?

- A. Create an execution plan by evaluating the difference between configuration file and state file.
- B. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.
- C. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.
- D. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.

**Answer: ([SHOW ANSWER](#))**

## NEW QUESTION: 82

When using constraint expressions to signify a version of a provider, which of the following are valid provider versions that satisfy the expression found in the following code snippet:

(select two)

1. terraform
2. {
3. required\_providers
4. {
5. aws = "~> 1.2.0"
6. }
7. }

- A. 1.3.1
- B. 1.2.3
- C. 1.2.9
- D. 1.3.0

**Answer: B,C** ([LEAVE A REPLY](#))

Explanation

As your Terraform usage becomes more advanced, there are some cases where you may need to modify the Terraform state. Rather than modify the state directly, the `terraform state` commands can be used in many cases instead. This command is a nested subcommand, meaning that it has further subcommands.

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

## NEW QUESTION: 83

Using multi-cloud and provider-agnostic tools provides which of the following benefits?

- A. Operations teams only need to learn and manage a single tool to manage infrastructure, regardless of where the infrastructure is deployed.
- B. Increased risk due to all infrastructure relying on a single tool for management.
- C. Can be used across major cloud providers and VM hypervisors.
- D. Slower provisioning speed allows the operations team to catch mistakes before they are applied.

**Answer: A,C** ([LEAVE A REPLY](#))

Explanation

Using a tool like Terraform can be advantageous for organizations deploying workloads across multiple public and private cloud environments. Operations teams only need to learn a single tool, single language, and can use the same tooling to enable a DevOps-like experience and workflows.

## NEW QUESTION: 84

```
resource "aws_s3_bucket" "example" { bucket = "my-test-s3-terraform-bucket" ...} resource  
"aws_iam_role"  
"test_role" { name = "test_role" ...}
```

Due to the way that the application code is written , the s3 bucket must be created before the test role is created , otherwise there will be a problem. How can you ensure that?

- A. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- B. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- C. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.
- D. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.

**Answer:** ([SHOW ANSWER](#))

Explanation

Use the depends\_on meta-argument to handle hidden resource dependencies that Terraform can't automatically infer.

Explicitly specifying a dependency is only necessary when a resource relies on some other resource's behavior but doesn't access any of that resource's data in its arguments.

## NEW QUESTION: 85

Which of the below command will upgrade the provider version to the latest acceptable one?

- A. terraform plan upgrade
- B. terraform provider -upgrade
- C. terraform init -upgrade
- D. terraform init -update

**Answer: C** ([LEAVE A REPLY](#))

Explanation

To upgrade to the latest acceptable version of each provider, run terraform init -upgrade. This command also upgrades to the latest versions of all Terraform modules.

<https://www.terraform.io/docs/configuration/providers.html>

## NEW QUESTION: 86

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

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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>

**NEW QUESTION: 87**

By default, a defined provisioner is a creation-time provisioner.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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

**NEW QUESTION: 88**

You want to use different AMI images for different regions and for the purpose you have defined following code block.

```
1.variable "images"
2.{ 
3. type = "map"
4.
5. default = {
6. us-east-1 = "image-1234"
7. us-west-2 = "image-4567"
8. us-west-1 = "image-4589"
9. }
10.}
```

What of the following approaches needs to be followed in order to select image-4589?

- A. var.images["us-west-1"]
- B. var.images[3]
- C. lookup(var.images["us-west-1"])
- D. var.images[2]

**Answer: A ([LEAVE A REPLY](#))****NEW QUESTION: 89**

Once a resource is marked as tainted, the next plan will show that the resource will be \_\_\_\_\_ and \_\_\_\_\_

\_\_\_\_\_ and the next apply will implement this change.

- A. recreated and tainted
- B. destroyed and recreated
- C. tainted and not destroyed

D. destroyed and not recreated

**Answer: B ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 90**

Please identify the offerings which are unique to Terraform Enterprise, and not available in either Terraform OSS, or Terraform Cloud. Select four.

- A. Audit Logs
- B. Private Network Connectivity
- C. VCS Integration
- D. Sentinel
- E. Clustering

**Answer: A,B,E ([LEAVE A REPLY](#))**

Explanation

<https://www.hashicorp.com/products/terraform/pricing/>

#### **NEW QUESTION: 91**

Which of the following state management command allow you to retrieve a list of resources that are part of the state file?

- A. terraform state list
- B. terraform state view
- C. terraform view
- D. terraform list

**Answer: ([SHOW ANSWER](#))**

Explanation

The terraform state list command is used to list resources within a Terraform state.

Usage: `terraform state list [options] [address...]`

The command will list all resources in the state file matching the given addresses (if any). If no addresses are given, all resources are listed.

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

#### **NEW QUESTION: 92**

A "backend" in Terraform determines how state is loaded and how an operation such as apply is executed.

Which of the following is not a supported backend type?

- A. Terraform enterprise
- B. Consul
- C. Github
- D. S3
- E. Artifactory

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Github is not a supported backend type.

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

### **NEW QUESTION: 93**

Which of the following Terraform commands will automatically refresh the state unless supplied with additional flags or arguments? Choose TWO correct answers.

- A. terraform validate
- B. terraform state
- C. terraform plan
- D. terraform output
- E. terraform apply

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 94**

Select the answer below that completes the following statement: Terraform Cloud can be managed from the CLI but requires \_\_\_\_\_?

- A. an API token
- B. a TOTP token
- C. a username and password
- D. authentication using MFA

**Answer: A ([LEAVE A REPLY](#))**

Explanation

API and CLI access are managed with API tokens, which can be generated in the Terraform Cloud UI. Each user can generate any number of personal API tokens, which allow access with their own identity and permissions. Organizations and teams can also generate tokens for automating tasks that aren't tied to an individual user.

### **NEW QUESTION: 95**

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. What command will do this?

- A. terraform taint

- B. `terraform apply`
- C. `terraform graph`
- D. `terraform refresh`

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially new IP address of a tainted server. The `plan` command will show this if this is the case.

This example will taint a single resource:

```
$ terraform taint aws_security_group.allow_all
```

The resource `aws_security_group.allow_all` in the module root has been marked as tainted.

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

## NEW QUESTION: 96

Select two answers to complete the following sentence: Before a new provider can be used, it must be \_\_\_\_\_ and \_\_\_\_\_.

- A. approved by HashiCorp
- B. uploaded to source control
- C. declared in the configuration
- D. initialized

**Answer: C,D ([LEAVE A REPLY](#))**

Explanation

Each time a new provider is added to configuration -- either explicitly via a provider block or by adding a resource from that provider -- Terraform must initialize the provider before it can be used. Initialization downloads and installs the provider's plugin so that it can later be executed.

## NEW QUESTION: 97

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

**Answer:** ([SHOW ANSWER](#))

Explanation

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>

### **NEW QUESTION: 98**

What are some of the features of Terraform state? (select three)

- A. increased performance
- B. determining the correct order to destroy resources
- C. inspection of cloud resources
- D. mapping configuration to real-world resources

**Answer:** A,D ([LEAVE A REPLY](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

# HashiCorp.TA-002-P.v2021-04-20.q98

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	98
<b>Version:</b>	v2021-04-20
<b># of views:</b>	2139
<b># of Questions views:</b>	105115

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2021-04-20.q98.html>

## NEW QUESTION: 1

Complete the following sentence:

For local state, the workspaces are stored directly in a \_\_\_\_\_.

- A. a file called terraform.tfstate.backup
- B. directory called terraform.workspaces.tfstate
- C. a file called terraform.tfstate
- D. directory called terraform.tfstate.d

**Answer:** ([SHOW ANSWER](#))

Explanation

For local state, Terraform stores the workspace states in a directory called terraform.tfstate.d.

<https://www.terraform.io/docs/state/workspaces.html#workspace-internals>

## NEW QUESTION: 2

A user has created three workspaces using the command line - prod, dev, and test. The user wants to create a fourth workspace named stage. Which command will the user execute to accomplish this?

- A. terraform workspace new stage
- B. terraform workspace -new stage
- C. terraform workspace -create stage
- D. terraform workspace create stage

**Answer:** A ([LEAVE A REPLY](#))

Explanation

The terraform workspace new command is used to create a new workspace.

<https://www.terraform.io/docs/commands/workspace/new.html>

## NEW QUESTION: 3

The Terraform language does not support user-defined functions, and so only the functions built in to the language are available for use.

- A. False
- B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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

#### **NEW QUESTION: 4**

When does terraform apply reflect changes in the cloud environment?

- A. None of the above
- B. After updating the state file
- C. However long it takes the resource provider to fulfill the request
- D. Immediately
- E. Based on the value provided to the -refresh command line argument

**Answer: A ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 5**

If you enable TF\_LOG = DEBUG, the log will be stored in syslog.log file in the current directory.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/internals/debugging.html>

#### **NEW QUESTION: 6**

Ric wants to enable detail logging and he wants highest verbosity of logs. Which of the following environment variable settings is correct option for him to select.

- A. Set TF\_LOG = DEBUG
- B. Set VAR\_TF = TRACE
- C. Set TF\_LOG = TRACE
- D. Set VAR\_TF\_LOG = TRACE

**Answer: ([SHOW ANSWER](#))**

Explanation

<https://www.terraform.io/docs/internals/debugging.html>

#### **NEW QUESTION: 7**

What Terraform feature is shown in the example below?

- A. conditional expression
- B. local values

C. dynamic block

D. data source

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 8**

Your team lead does not trust the junior terraform engineers who now have access to the git repo . So , he wants you to have some sort of a checking layer , whereby , you can ensure that the juniors will not create any non-compliant resources that might lead to a security audit failure in future. What can you do to efficiently enforce this?

A. Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.

B. Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them.

C. Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code.

D. Create a git master branch , and implement PR . Every change needs to be reviewed by you , before being merged to the master branch.

**Answer:** B ([LEAVE A REPLY](#))

Explanation

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### **NEW QUESTION: 9**

Command terraform refresh will update state file?

A. False

B. True

**Answer:** B ([LEAVE A REPLY](#))

Explanation

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. If the state is changed, this may cause changes to occur during the next plan or apply.

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

### **NEW QUESTION: 10**

After running into issues with Terraform, you need to enable verbose logging to assist with troubleshooting the error. Which of the following values provides the MOST verbose logging?

- A. ERROR
- B. INFO
- C. WARN
- D. TRACE
- E. DEBUG

**Answer:** D ([LEAVE A REPLY](#))

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

Examples:

```
export TF_LOG=DEBUG  
export TF_LOG=TRACE
```

### **NEW QUESTION: 11**

Terraform works well in Windows but a Windows server is required.

- A. False
- B. True

**Answer:** A ([LEAVE A REPLY](#))

Explanation

You may see this question in actual exam. Please remember : Terraform does not require GO language to be installed as a prerequisite and it does not require a Windows Server as well.

### **NEW QUESTION: 12**

terraform state subcommands such as list are read-only commands, do read-only commands create state backup files?

- A. Yes
- B. No

**Answer:** ([SHOW ANSWER](#))

Explanation

Subcommands that are read-only (such as list) do not write any backup files since they aren't modifying the state.

All terraform state subcommands that modify the state write backup files. The path of these backup file can be controlled with -backup.

<https://www.terraform.io/docs/commands/state/index.html#backups>

### **NEW QUESTION: 13**

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

**Answer: ([SHOW ANSWER](#))**

Explanation

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>

### **NEW QUESTION: 14**

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

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 15**

By default, where does Terraform store its state file?

- A. Amazon S3 bucket
- B. shared directory
- C. remotely using Terraform Cloud
- D. current working directory

**Answer: ([SHOW ANSWER](#))**

Explanation

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

### **NEW QUESTION: 16**

A single terraform resource file that defines an aws\_instance resource can simply be renamed to vsphere\_virtual\_machine in order to switch cloud providers.

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Every provider has its own required and allowed declarations none of which match between cloud providers.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF** **Special Discount: freecram**)

### NEW QUESTION: 17

Which of the following represents a feature of Terraform Cloud that is NOT free to customers?

- A. Roles and Team Management
- B. WorkSpace Management
- C. Private Module Registry
- D. VCS Integration

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Role Based Access Controls (RBAC) for controlling permissions for who has access to what configurations within an organization and it is not free to customers.

<https://www.hashicorp.com/products/terraform/pricing/>

### NEW QUESTION: 18

Your organization has moved to AWS and has manually deployed infrastructure using the console. Recently, a decision has been made to standardize on Terraform for all deployments moving forward.

What can you do to ensure that all existing is managed by Terraform moving forward without interruption to existing services?

- A. Submit a ticket to AWS and ask them to export the state of all existing resources and use terraform import to import them into the state file.
- B. Delete the existing resources and recreate them using new a Terraform configuration so Terraform can manage them moving forward.
- C. Resources that are manually deployed in the AWS console cannot be imported by Terraform.
- D. Using terraform import, import the existing infrastructure into your Terraform state.

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

## NEW QUESTION: 19

Matt wants to import a manually created EC2 instance into terraform so that he can manage the EC2 instance through terraform going forward. He has written the configuration file of the EC2 instance before importing it to Terraform. Following is the code:

```
resource "aws_instance" "matt_ec2" { ami = "ami-bg2640de" instance_type = "t2.micro"  
vpc_security_group_ids = ["sg-6ae7d613", "sg-53370035"] key_name = "mysecret"  
subnet_id =  
"subnet-9e3cfbc5" }
```

The instance id of that EC2 instance is i-0260835eb7e9bd40 How he can import data of EC2 to state file?

- A. `terraform import aws_instance.id = i-0260835eb7e9bd40`
- B. `terraform import i-0260835eb7e9bd40`
- C. `terraform import aws_instance.i-0260835eb7e9bd40`
- D. `terraform import aws_instance.matt_ec2 i-0260835eb7e9bd40`

**Answer: D ([LEAVE A REPLY](#))**

Explanation

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

## NEW QUESTION: 20

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

**Answer: D ([LEAVE A REPLY](#))**

## NEW QUESTION: 21

Which of the following Terraform files should be ignored by Git when committing code to a repo? (select Three)

- A. Files named exactly `terraform.tfvars` or `terraform.tfvars.json`.
- B. Any files with names ending in `.auto.tfvars` or `.auto.tfvars.json`.
- C. `input.tf`
- D. `terraform.tfstate`
- E. `output.tf`

**Answer: A,B,D ([LEAVE A REPLY](#))**

Explanation

The `.gitignore` file should be configured to ignore Terraform files that either contain sensitive data or are not required to save.

Terraform state (`terraform.tfstate`) can contain sensitive data, depending on the resources in use and your definition of "sensitive." The state contains resource IDs and all resource attributes. For resources such as databases, this may contain initial passwords.

When using local state, state is stored in plain-text JSON files.

The `terraform.tfvars` file may contain sensitive data, such as passwords or IP addresses of an environment that you may not want to share with others.

## NEW QUESTION: 22

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

**Answer:** A ([LEAVE A REPLY](#))

### **NEW QUESTION: 23**

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 = var.num\_servers
- B.** servers = variable.num\_servers
- C.** servers = num\_servers
- D.** servers = var(num\_servers)

**Answer:** C ([LEAVE A REPLY](#))

### **NEW QUESTION: 24**

In Terraform Enterprise, a workspace can be mapped to how many VCS repos?

- A.** 5
- B.** 2
- C.** 3
- D.** 1

**Answer:** D ([LEAVE A REPLY](#))

Explanation

A workspace can only be configured to a single VCS repo, however, multiple workspaces can use the same repo.

<https://www.terraform.io/docs/cloud/workspaces/vcs.html>

### **NEW QUESTION: 25**

State is a requirement for Terraform to function

- A.** True
- B.** False

**Answer:** ([SHOW ANSWER](#))

Explanation

State is a necessary requirement for Terraform to function. It is often asked if it is possible for Terraform to work without state, or for Terraform to not use state and just inspect cloud resources on every run.

Purpose of Terraform State

State is a necessary requirement for Terraform to function. It is often asked if it is possible for Terraform to work without state, or for Terraform to not use state and just inspect cloud resources on every run. This page will help explain why Terraform state is required.

As you'll see from the reasons below, state is required. And in the scenarios where Terraform may be able to get away without state, doing so would require shifting massive amounts of complexity from one place (state) to another place (the replacement concept).

## 1. Mapping to the Real World

Terraform requires some sort of database to map Terraform config to the real world. When you have a resource resource "aws\_instance" "foo" in your configuration, Terraform uses this map to know that instance i- abcd1234 is represented by that resource.

For some providers like AWS, Terraform could theoretically use something like AWS tags. Early prototypes of Terraform actually had no state files and used this method. However, we quickly ran into problems. The first major issue was a simple one: not all resources support tags, and not all cloud providers support tags.

Therefore, for mapping configuration to resources in the real world, Terraform uses its own state structure.

## 2. Metadata

Alongside the mappings between resources and remote objects, Terraform must also track metadata such as resource dependencies.

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

One way to avoid this would be for Terraform to know a required ordering between resource types. For example, Terraform could know that servers must be deleted before the subnets they are a part of. The complexity for this approach quickly explodes, however: in addition to Terraform having to understand the ordering semantics of every resource for every cloud, Terraform must also understand the ordering across providers.

Terraform also stores other metadata for similar reasons, such as a pointer to the provider configuration that was most recently used with the resource in situations where multiple aliased providers are present.

## 3. Performance

In addition to basic mapping, Terraform stores a cache of the attribute values for all resources in the state. This is the most optional feature of Terraform state and is done only as a performance improvement.

When running a terraform plan, Terraform must know the current state of resources in order to effectively determine the changes that it needs to make to reach your desired configuration.

For small infrastructures, Terraform can query your providers and sync the latest attributes from all your resources. This is the default behavior of Terraform: for every plan and apply, Terraform will sync all resources in your state.

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to query multiple resources at once, and the round trip time for each resource is hundreds of milliseconds. On top of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

#### 4. Syncing

In the default configuration, Terraform stores the state in a file in the current working directory where Terraform was run. This is okay for getting started, but when using Terraform in a team it is important for everyone to be working with the same state so that operations will be applied to the same remote objects.

Remote state is the recommended solution to this problem. With a fully-featured state backend, Terraform can use remote locking as a measure to avoid two or more different users accidentally running Terraform at the same time, and thus ensure that each Terraform run begins with the most recent updated state.

### NEW QUESTION: 26

Valarie has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code. Valarie wants to hide the output value in the CLI after terraform apply that's why she has used sensitive parameter.

```
1. output "db_password" {  
2.   value = local.db_password  
3.   sensitive = true  
4. }
```

Since sensitive is set to true, will the value associated with db password be available in plain-text in the state file for everyone to read?

- A. Yes
- B. No

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Outputs can be marked as containing sensitive material by setting the `sensitive` attribute to true, like this:

```
output "sensitive" {  
  sensitive = true  
  value = VALUE  
}
```

When outputs are displayed on-screen following a `terraform apply` or `terraform refresh`, sensitive outputs are redacted, with `<sensitive>` displayed in place of their value.

Limitations of Sensitive Outputs

The values of sensitive outputs are still stored in the Terraform state, and available using the `terraform output` command, so cannot be relied on as a sole means of protecting values.

Sensitivity is not tracked internally, so if the output is interpolated in another module into a resource, the value will be displayed.

### **NEW QUESTION: 27**

Select two answers to complete the following sentence: Before a new provider can be used, it must be \_\_\_\_\_ and \_\_\_\_\_.

- A. approved by HashiCorp
- B. uploaded to source control
- C. declared in the configuration
- D. initialized

**Answer:** ([SHOW ANSWER](#))

Explanation

Each time a new provider is added to configuration -- either explicitly via a provider block or by adding a resource from that provider -- Terraform must initialize the provider before it can be used. Initialization downloads and installs the provider's plugin so that it can later be executed.

### **NEW QUESTION: 28**

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

**Answer:** E ([LEAVE A REPLY](#))

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

### **NEW QUESTION: 29**

A "backend" in Terraform determines how state is loaded and how an operation such as `apply` is executed.

Which of the following is not a supported backend type?

- A. Terraform enterprise
- B. Consul
- C. Github
- D. S3
- E. Artifactory

**Answer:** C ([LEAVE A REPLY](#))

## Explanation

Github is not a supported backend type.

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

## NEW QUESTION: 30

What is a downside to using the Vault provider to read secrets from Vault?

- A. Secrets are persisted to the state file and plans.
- B. Terraform and Vault must be running on the same version.
- C. Terraform and Vault must be running on the same physical host.
- D. Terraform requires a unique auth method to work with Vault.

**Answer: A ([LEAVE A REPLY](#))**

## Explanation

The Vault provider allows Terraform to read from, write to, and configure Hashicorp Vault. Interacting with Vault from Terraform causes any secrets that you read and write to be persisted in both Terraform's state file and in any generated plan files. For any Terraform module that reads or writes Vault secrets, these files should be treated as sensitive and protected accordingly.

## NEW QUESTION: 31

What is the provider for this fictitious resource?

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

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

**Answer: C ([LEAVE A REPLY](#))**

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**

## **NEW QUESTION: 32**

Your manager has instructed you to start using terraform for the entire infra provisioning of the application stack. There are 4 environments - DEV , QA , UAT , and PROD. The application team has asked for complete segregation between these environments including the backend , state , and also configurations ,since there will be unique resources in different environments . What is the possible way to structure the terraform code to facilitate that.

- A.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to a different backend.
- B.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to the same backend.
- C.** Implement terraform workspaces , and map each environment with one workspace.
- D.** Enable remote backend storage . Configure 4 different backend storages , one for each environment.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

In particular, organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

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

## **NEW QUESTION: 33**

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

- A.** True
- B.** False

**Answer: B ([LEAVE A REPLY](#))**

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

## **NEW QUESTION: 34**

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A.** Explicit use of a provider block in configuration, optionally including a version constraint.
- B.** Use of any resource belonging to a particular provider in a resource or data block in configuration.
- C.** Existence of any resource instance belonging to a particular provider in the current state.

D. Existence of any provider plugins found locally in the working directory.

**Answer: A,B,C ([LEAVE A REPLY](#))**

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider dependency. The plugin can exist without any reference to it in the terraform configuration.

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

### **NEW QUESTION: 35**

What are some of the problems of how infrastructure was traditionally managed before Infrastructure as Code?

(select three)

- A. Requests for infrastructure or hardware required a ticket, increasing the time required to deploy applications
- B. Traditional deployment methods are not able to meet the demands of the modern business where resources tend to live days to weeks, rather than months to years
- C. Traditionally managed infrastructure can't keep up with cyclic or elastic applications
- D. Pointing and clicking in a management console is a scalable approach and reduces human error as businesses are moving to a multi-cloud deployment model

**Answer: A,B,C ([LEAVE A REPLY](#))**

Explanation

Businesses are making a transition where traditionally-managed infrastructure can no longer meet the demands of today's businesses. IT organizations are quickly adopting the public cloud, which is predominantly API-driven. To meet customer demands and save costs, application teams are architecting their applications to support a much higher level of elasticity, supporting technology like containers and public cloud resources.

These resources may only live for a matter of hours; therefore the traditional method of raising a ticket to request resources is no longer a viable option. Pointing and clicking in a management console is NOT scale and increases the chance of human error.

### **NEW QUESTION: 36**

Consider the following Terraform 0.12 configuration snippet:

```
1. variable "vpc_cidrs" {  
2.   type = map  
3.   default = {  
4.     us-east-1 = "10.0.0.0/16"  
5.     us-east-2 = "10.1.0.0/16"  
6.     us-west-1 = "10.2.0.0/16"  
7.     us-west-2 = "10.3.0.0/16"  
8.   }  
9. }
```

10.  
11. resource "aws\_vpc" "shared" {  
12. cidr\_block = \_\_\_\_\_  
13. }

How would you define the cidr\_block for us-east-1 in the aws\_vpc resource using a variable?

- A. var.vpc\_cidrs[0]
- B. var.vpc\_cidrs["us-east-1"]
- C. var.vpc\_cidrs.0
- D. vpc\_cidrs["us-east-1"]

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 37**

Which of the following challenges would Terraform be a candidate for solving? (Select THREE)

- A. Have a single interoperable tool to manage the variety of services including GitHub repositories, MySQL database, and Kubernetes clusters.
- B. Reduce the number of workflows needed for managing infrastructure across each of the companies public and private clouds.
- C. Enable self-service infrastructure to allocate resources on your proprietary private cloud.
- D. Utilize a single tool for all of the infrastructure and configuration management needs.

**Answer: A,B,C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 38**

lookup retrieves the value of a single element from which of the below data type?

- A. map
- B. set
- C. string
- D. list

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

### **NEW QUESTION: 39**

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

- 1. max
- 2. min
- 3. join
- 4. replace
- 5. list

6. length
  7. range
- A. Backends
  - B. Data sources
  - C. Named values
  - D. Functions

**Answer: D ([LEAVE A REPLY](#))**

Explanation

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not support user-defined functions, and only the functions built into the language are available for use.

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

#### **NEW QUESTION: 40**

Which of the following are string functions? Select three

- A. tostring
- B. tonumber
- C. Chomp
- D. format
- E. join

**Answer: C,D,E ([LEAVE A REPLY](#))**

Explanation

tonumber and tostring are Type Conversion function

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

#### **NEW QUESTION: 41**

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

**Answer: B ([LEAVE A REPLY](#))**

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

#### **NEW QUESTION: 42**

When should you use the force-unlock command?

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

D. Your apply failed due to a state lock

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Manually unlock the state for the defined configuration.

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

### **NEW QUESTION: 43**

A user creates three workspaces from the command line - prod, dev, and test. Which of the following commands will the user run to switch to the dev workspace?

- A. terraform workspace dev
- B. terraform workspace select dev
- C. terraform workspace -switch dev
- D. terraform workspace switch dev

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The terraform workspace select command is used to choose a different workspace to use for further operations.

<https://www.terraform.io/docs/commands/workspace/select.html>

### **NEW QUESTION: 44**

Your company has been using Terraform Cloud for a some time now . But every team is creating their own modules , and there is no standardization of the modules , with each team creating the resources in their own unique way . You want to enforce a standardization of the modules across the enterprise . What should be your approach.

- A. Create individual workspaces for each team , and ask them to share modules across workspaces.
- B. Implement a Private module registry in Terraform cloud , and ask teams to reference them.
- C. Upgrade to Terraform enterprise , since this is not possible in terraform cloud.
- D. Upload the modules in the terraform public module registry , and ask teams to reference them

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Terraform Cloud's private module registry helps you share Terraform modules across your organization. It includes support for module versioning, a searchable and filterable list of available modules, and a configuration designer to help you build new workspaces faster. By design, the private module registry works much like the public Terraform Registry. If you're already used the public registry, Terraform Cloud's registry will feel familiar.

Understand the different offerings in Terraform OS, Terraform Cloud and Terraform Enterprise. Terraform Cloud's private module registry helps you share Terraform modules across your organization.

<https://www.terraform.io/docs/cloud/registry/index.html>

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

### NEW QUESTION: 45

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. Terraform attempts to provision the resource up to three times before exiting with an error
- B. the terraform plan is rolled back and all provisioned resources are removed
- C. it is automatically deleted
- D. the resource is marked as tainted

**Answer:** D ([LEAVE A REPLY](#))

Explanation

If a resource successfully creates but fails during provisioning, Terraform will error and mark the resource as

"tainted". A resource that is tainted has been physically created, but can't be considered safe to use since provisioning failed. Terraform also does not automatically roll back and destroy the resource during the apply when the failure happens, because that would go against the execution plan: the execution plan would've said a resource will be created, but does not say it will ever be deleted.

### NEW QUESTION: 46

Which of the following is considered a Terraform plugin?

- A. Terraform language
- B. Terraform tooling
- C. Terraform logic
- D. Terraform provider

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

<https://www.terraform.io/docs/plugins/basics.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

### **NEW QUESTION: 47**

Only the user that generated a plan may apply it.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The optional -out argument can be used to save the generated plan to a file for later execution with terraform apply, which can be useful when running Terraform in automation.

Reference: <https://learn.hashicorp.com/tutorials/terraform/automate-terraform>

### **NEW QUESTION: 48**

Which of the following statements best describes the Terraform list(...) type?

- A. a collection of values where each is identified by a string label.
- B. a sequence of values identified by consecutive whole numbers starting with zero.
- C. a collection of unique values that do not have any secondary identifiers or ordering.
- D. a collection of named attributes that each have their own type.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

A terraform list is a sequence of values identified by consecutive whole numbers starting with zero.

<https://www.terraform.io/docs/configuration/types.html#structural-types>

### **NEW QUESTION: 49**

What kind of resource dependency is stored in terraform.tfstate file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 50**

In the example below, where is the value of the DNS record's IP address originating from?

```
1. resource "aws_route53_record" "www"
2. {
3.   zone_id = aws_route53_zone.primary.zone_id
4.   name = "www.example.com"
5.   type = "A"
6.   ttl = "300"
7.   records = [module.web_server.instance_ip_address]
8. }
```

- A. The regular expression named module.web\_server
- B. The output of a module named web\_server
- C. By querying the AWS EC2 API to retrieve the IP address
- D. Value of the web\_server parameter from the variables.tf file

**Answer:** ([SHOW ANSWER](#))

Explanation

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\_address, you could access that value as module.web\_server.instance\_ip\_address.

## NEW QUESTION: 51

HashiCorp offers multiple versions of Terraform, including Terraform open-source, Terraform Cloud, and Terraform Enterprise. Which of the following Terraform features are only available in the Enterprise edition?

(select four)

- A. SAML/SSO
- B. Sentinel
- C. Audit Logs
- D. Clustering
- E. Private Module Registry
- F. Private Network Connectivity

**Answer:** A,C,F ([LEAVE A REPLY](#))

Explanation

While there are a ton of features that are available to open source users, many features that are part of the Enterprise offering are geared towards larger teams and enterprise functionality. To see what specific features are part of Terraform Cloud and Terraform Enterprise, check out this link.

<https://www.hashicorp.com/products/terraform/pricing/>

## NEW QUESTION: 52

True or False? Each Terraform workspace uses its own state file to manage the infrastructure associated with that particular workspace.

- A. False
- B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The persistent data stored in the backend belongs to a workspace. Initially, the backend has only one workspace, called "default", and thus there is only one Terraform state associated with that configuration.

### **NEW QUESTION: 53**

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

**Answer: C ([LEAVE A REPLY](#))**

Explanation

terraform workspace list

The command will list all existing workspaces.

Reference: <https://www.terraform.io/docs/cli/commands/workspace/list.html>

### **NEW QUESTION: 54**

Select all features which are exclusive to Terraform Enterprise. (Select Three)

- A. Sentinel
- B. Cost Estimation
- C. Audit Logs
- D. Clustering
- E. SAML/SSO

**Answer: C,D,E ([LEAVE A REPLY](#))**

Explanation

Sentinel and Cost Estimation are also available in Terraform Cloud

<https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 55**

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

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

**Answer: D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 56**

What does the command terraform fmt do?

- A. Rewrite Terraform configuration files to a canonical format and style.
- B. Deletes the existing configuration file.
- C. Updates the font of the configuration file to the official font supported by HashiCorp.
- D. Formats the state file in order to ensure the latest state of resources can be obtained.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The terraform fmt command is used to rewrite Terraform configuration files to a canonical format and style.

This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability.

Other Terraform commands that generate Terraform configuration will produce configuration files that conform to the style imposed by terraform fmt, so using this style in your own files will ensure consistency.

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

### **NEW QUESTION: 57**

Terraform providers are always installed from the Internet.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

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>

### **NEW QUESTION: 58**

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

## NEW QUESTION: 59

When configuring a remote backend in Terraform, it might be a good idea to purposely omit some of the required arguments to ensure secrets and other important data aren't inadvertently shared with others. What are the ways the remaining configuration can be added to Terraform so it can initialize and communicate with the backend? (select three)

- A. directly querying HashiCorp Vault for the secrets
- B. command-line key/value pairs
- C. use the `-backend-config=PATH` to specify a separate config file
- D. interactively on the command line

**Answer: B,C,D ([LEAVE A REPLY](#))**

Explanation

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process. There are several ways to supply the remaining arguments:

<https://www.terraform.io/docs/backends/init.html#backend-initialization>

## NEW QUESTION: 60

Where does the Terraform local backend store its state?

- A. In the `/tmp` directory
- B. In the `terraform.tfvars` file
- C. In the `terraform.tfstate` file
- D. In the user's `.terraformrc` file

**Answer: ([SHOW ANSWER](#))**

Explanation

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>

## NEW QUESTION: 61

What does terraform refresh command do?

- A. terraform refresh syncs the state file with the real world infrastructure.
- B. terraform refresh command basically updates the configuration file with the current state of the actual infrastructure
- C. terraform refresh is used to change/modify the infrastructure based on the existing state file, at that moment.
- D. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- E. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

## NEW QUESTION: 62

True or False? By default, Terraform destroy will prompt for confirmation before proceeding.

- A. False

- B. True

**Answer:** B ([LEAVE A REPLY](#))

## NEW QUESTION: 63

Once a resource is marked as tainted, the next plan will show that the resource will be \_\_\_\_\_ and \_\_\_\_\_

\_\_\_\_\_ and the next apply will implement this change.

- A. tainted and not destroyed
- B. destroyed and recreated
- C. destroyed and not recreated
- D. recreated and tainted

**Answer:** B ([LEAVE A REPLY](#))

## NEW QUESTION: 64

Refer to the following terraform variable definition

```
variable "track_tag" { type = list default = ["data_ec2","integration_ec2","digital_ec2"]}  
track_tag = { Name = element(var.track_tag,count.index)} If count.index is set to 2, which  
of the following values will be assigned to the name attribute of track_tag variable?
```

- A. data\_ec2
- B. track\_tag
- C. integration\_ec2
- D. digital\_ec2

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 65**

You want to get involved in the development of Terraform. As this is an open source project, you would like to contribute a fix for an open issue of Terraform. What programming language will need to use to write the fix?

- A. It depends on which command issue related to.
- B. Python
- C. Go
- D. Java

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Basic programming knowledge. Terraform and Terraform Plugins are written in the Go programming language, but even if you've never written a line of Go before, you're still welcome to take a dive into the code and submit patches. The community is happy to assist with code reviews and offer guidance specific to Go.

### **NEW QUESTION: 66**

Talal is a DevOps engineer and he has deployed the production infrastructure using Terraform. He is using a very large configuration file to maintain and update the actual infrastructure. As the infrastructure have grown to a very complex and large, he has started experiencing slowness when he runs terraform plan. What are the options for him to resolve this slowness?

- A. Use -refresh=true flag as well as the -target flag with terraform plan in order to work around this.
- B. Run terraform refresh every time before running terraform plan.
- C. Break large configurations into several smaller configurations that can each be independently applied.
- D. Use -refresh=false flag as well as the -target flag with terraform plan in order to work around this.

**Answer: C,D ([LEAVE A REPLY](#))**

Explanation

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to query multiple resources at once, and the round trip time for each

resource is hundreds of milliseconds. On top of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

Although 'Use `-refresh=false` flag as well as the `-target` flag with terraform plan in order to work around this.' is a solution, but its not always recommended. Instead of using `-target` as a means to operate on isolated portions of very large configurations, prefer instead to break large configurations into several smaller configurations that can each be independently applied. Data sources can be used to access information about resources created in other configurations, allowing a complex system architecture to be broken down into more manageable parts that can be updated independently.

Option 'Run terraform refresh every time before running terraform plan.' and 'Use `-refresh=true` flag as well as the `-target` flag with terraform plan in order to work around this.' is not correct because in both the cases terraform will query every resources of the infrastructure.

### **NEW QUESTION: 67**

What is not processed when running a terraform refresh?

- A. State file
- B. Configuration file
- C. Credentials
- D. Cloud provider

**Answer: C,D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 68**

By default, provisioners that fail will also cause the Terraform apply itself to error. How can you change this default behavior within a provisioner?

- A. provisioner "local-exec" { on\_failure = "next" }
- B. provisioner "local-exec" { when = "failure" terraform apply }
- C. provisioner "local-exec" { on\_failure = "continue" }
- D. provisioner "local-exec" { on\_failure = continue }

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 69**

What does terraform plan do ?

- A. Create an execution plan by evaluating the difference between configuration file and state file.

- B. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.
- C. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.
- D. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.

**Answer:** A ([LEAVE A REPLY](#))

#### **NEW QUESTION: 70**

Which of the following Terraform commands will automatically refresh the state unless supplied with additional flags or arguments? Choose TWO correct answers.

- A. `terraform state`
- B. `terraform output`
- C. `terraform validate`
- D. `terraform plan`
- E. `terraform apply`

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 71**

You have created two workspaces PROD and DEV. You have switched to DEV and provisioned DEV infrastructure from this workspace. Where is your state file stored?

- A. `terraform.d`
- B. `terraform.tfstate`
- C. `terraform.tfstate.DEV`
- D. `terraform.tfstate.d`

**Answer:** D ([LEAVE A REPLY](#))

Explanation

Terraform stores the workspace states in a directory called `terraform.tfstate.d`. This directory should be treated similarly to default workspace state file `terraform.tfstate` `main.tf` provider.`tf` `terraform.tfstate.d` DEV `terraform.tfstate` # DEV workspace state file PROD `terraform.tfstate` # PROD workspace state file `terraform.tfvars` # Default workspace state file `variables.tf`

#### **NEW QUESTION: 72**

The following is a snippet from a Terraform configuration file:

Which, when validated, results in the following error:

Fill in the blank in the error message with the correct string from the list below.

- A. `version`
- B. `multi`
- C. `label`
- D. `alias`

**Answer: D ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-providerinstances>

### **NEW QUESTION: 73**

Why might a user opt to include the following snippet in their configuration file?

- A. Terraform 0.12 introduced substantial changes to the syntax used to write Terraform configuration
- B. The user wants to ensure that the application being deployed is a minimum version of 0.12
- C. this ensures that all Terraform providers are above a certain version to match the application being deployed
- D. versions before Terraform 0.12 were not approved by HashiCorp to be used in production

**Answer: A ([LEAVE A REPLY](#))**

### **NEW QUESTION: 74**

What is the default backend for Terraform?

- A. consul
- B. gcs
- C. local
- D. etcd

**Answer: ([SHOW ANSWER](#))**

Explanation

By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to.

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

### **NEW QUESTION: 75**

Which task does terraform init not perform?

- 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

**Answer: D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 76**

When Terraform needs to be installed in a location where it does not have internet access to download the installer and upgrades, the installation is generally known as to be \_\_\_\_\_.

- A. a private install
- B. disconnected
- C. air-gapped
- D. non-traditional

**Answer:** ([SHOW ANSWER](#))

Explanation

A Terraform Enterprise install that is provisioned on a network that does not have Internet access is generally known as an air-gapped install. These types of installs require you to pull updates, providers, etc. from external sources vs. being able to download them directly.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 77

Which of the below features of Terraform can be used for managing small differences between different environments which can act more like completely separate working directories.

- A. Repositories
- B. Workspaces
- C. Environment Variables
- D. Backends

**Answer:** B ([LEAVE A REPLY](#))

Explanation

workspaces allow conveniently switching between multiple instances of a single configuration within its single backend. They are convenient in a number of situations, but cannot solve all problems.

A common use for multiple workspaces is to create a parallel, distinct copy of a set of infrastructure in order to test a set of changes before modifying the main production infrastructure. For example, a developer working on a complex set of infrastructure changes might create a new temporary workspace in order to freely experiment with changes without affecting the default workspace.

Non-default workspaces are often related to feature branches in version control. The default workspace might correspond to the "master" or "trunk" branch, which describes the intended state of production infrastructure.

When a feature branch is created to develop a change, the developer of that feature might create a corresponding workspace and deploy into it a temporary "copy" of the main infrastructure so that changes can be tested without affecting the production infrastructure. Once the change is merged and deployed to the default workspace, the test infrastructure can be destroyed and the temporary workspace deleted.

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

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

## NEW QUESTION: 78

Which of the following best describes the default local backend?

- A. The local backend is where Terraform Enterprise stores logs to be processed by an log collector.
- B. The local backend stores state on the local filesystem, locks the state using system APIs, and performs operations locally.
- C. The local backend is the directory where resources deployed by Terraform have direct access to in order to update their current state.
- D. The local backend is how Terraform connects to public cloud services, such as AWS, Azure, or GCP.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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

```
terraform {  
  backend "local" {  
    path = "relative/path/to/terraform.tfstate"  
  }  
}
```

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

## NEW QUESTION: 79

Your team has started using terraform OSS in a big way , and now wants to deploy multi region deployments (DR) in aws using the same terraform files . You want to deploy the same infra (VPC,EC2 ...) in both us-east-1 ,and us-west-2 using the same script , and then peer the VPCs across both the regions to enable DR traffic. But , when you run your script , all resources are getting created in only the default provider region.

What should you do? Your provider setting is as below -

```
# The default provider configuration provider "aws" { region = "us-east-1" }
```

- A.** No way to enable this via a single script . Write 2 different scripts with different default providers in the  
2 scripts , one for us-east , another for us-west.
- B.** Create a list of regions , and then use a for-each to iterate over the regions , and create the same resources ,one after the one , over the loop.
- C.** Use provider alias functionality , and add another provider for us-west region . While creating the resources using the tf script , reference the appropriate provider (using the alias).
- D.** Manually create the DR region , once the Primary has been created , since you are using terraform OSS , and multi region deployment is only available in Terraform Enterprise.

**Answer: C ([LEAVE A REPLY](#))**

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
alias = "west"
region = "us-west-2"
}
```

<https://www.terraform.io/docs/configuration/providers.html>

## NEW QUESTION: 80

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A.** TF\_TRACE
- B.** TF\_DEBUG
- C.** TF\_LOG
- D.** TF\_INFO

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

## NEW QUESTION: 81

Select the operating systems which are supported for a clustered Terraform Enterprise:

(select four)

- A. Unix
- B. Red Hat
- C. CentOS
- D. Amazon Linux
- E. Ubuntu

**Answer: B,C,D,E ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/enterprise/before-installing/index.html#operating-systemrequirements>

## NEW QUESTION: 82

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address aws\_instance.web refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.
- C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.
- D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer: ([SHOW ANSWER](#))**

Explanation

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:  
module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

`aws_instance.web[3]`

Refers to only the last instance in the config, and an address like this:

`aws_instance.web`

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

### **NEW QUESTION: 83**

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

**Answer: A,B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 84**

Which of the following variable definition files will terraform load automatically?

- A. `terraform.tfvar`
- B. Any files with names ending in `.auto.tfvars.json`
- C. `terraform.tfvars`
- D. `terraform.tfvars.json`

**Answer: B,C,D ([LEAVE A REPLY](#))**

Explanation

Terraform also automatically loads a number of variable definitions files if they are present:

Files named exactly `terraform.tfvars` or `terraform.tfvars.json`.

Any files with names ending in `.auto.tfvars` or `.auto.tfvars.json`.

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

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

### **NEW QUESTION: 85**

What is the standard workflow that a developer follows while working with terraform open source version?

- A. Run terraform refresh to update the terraform state , then write the terraform code , and finally run terraform apply.
- B. Run terraform destroy first since you need to start from fresh every time , before running terraform apply.
- C. Write terraform code , and run terraform push , to update the terraform state to the remote repo , which in turn will take care of the next steps.
- D. Write the terraform code on the developer machine , run terraform plan to check the changes , and run terraform apply to provision the infra.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

You do not need to run terraform refresh as terraform plan implicitly will run terraform refresh.

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

### **NEW QUESTION: 86**

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

**Answer: B,D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 87**

Which feature of Terraform allows multiple state files for a single configuration file depending upon the environment?

- A. Terraform Enterprise
- B. Terraform Remote Backends
- C. Terraform Modules
- D. Terraform Workspaces

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 88**

You have created a custom variable definition file testing.tfvars. How will you use it for provisioning infrastructure?

- A. `terraform apply -var-state-file ="testing.tfvars"`
- B. `terraform plan -var-file="testing.tfvar"`

- C. `terraform apply -var-file="testing.tfvars"`
- D. `terraform apply var-file="testing.tfvars"`

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 89**

Which of the below configuration file formats are supported by Terraform? (Select TWO)

- A. Node
- B. JSON
- C. Go
- D. YAML
- E. HCL

**Answer: B,E ([LEAVE A REPLY](#))**

Explanation

Terraform supports both HashiCorp Configuration Language (HCL) and JSON formats for configurations.

<https://www.terraform.io/docs/configuration/>

### **NEW QUESTION: 90**

You have configured an Auto Scaling group in AWS to automatically scale the number of instances behind a load balancer based on the instances CPU utilization. The instances are configured using a Launch Configuration. You have observed that the Auto Scaling group doesn't successfully scale when you apply changes that require replacing the Launch Configuration. Why is this happening?

- A. You need to configure an explicit dependency for the Auto Scaling group using the `depends_on` meta-parameter.
- B. You need to configure an explicit dependency for the Launch Configuration using the `depends_on` meta-parameter.
- C. You need to configure the Auto Scaling group's `create_before_destroy` meta-parameter.
- D. You need to configure the Launch Configuration's `create_before_destroy` meta-parameter.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

[https://www.terraform.io/docs/providers/aws/r/launch\\_configuration.html#using-withautoscaling-groups](https://www.terraform.io/docs/providers/aws/r/launch_configuration.html#using-withautoscaling-groups)

### **NEW QUESTION: 91**

Terraform Enterprise (also referred to as pTFE) requires what type of backend database for a clustered deployment?

- A. PostgreSQL

- B. Cassandra
- C. MySQL
- D. MSSQL

**Answer:** ([SHOW ANSWER](#))

Explanation

External Services mode stores the majority of the stateful data used by the instance in an external PostgreSQL database and an external S3-compatible endpoint or Azure blob storage. There is still critical data stored on the instance that must be managed with snapshots. Be sure to check the PostgreSQL Requirements for information that needs to be present for Terraform Enterprise to work. This option is best for users with expertise managing PostgreSQL or users that have access to managed PostgreSQL offerings like AWS RDS.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

## NEW QUESTION: 92

While Terraform is generally written using the HashiCorp Configuration Language (HCL), what other syntax can Terraform be expressed in?

- A. JSON
- B. YAML
- C. TypeScript
- D. XML

**Answer:** A ([LEAVE A REPLY](#))

Explanation

The constructs in the Terraform language can also be expressed in JSON syntax, which is harder for humans to read and edit but easier to generate and parse programmatically.

## NEW QUESTION: 93

You have created an AWS EC2 instance of type t2.micro through your terraform configuration file ec2.tf .

Now you want to change the instance type from t2.micro to t2.medium. Accordingly you have changed your configuration file and ran terraform plan. After running terraform plan you check the output and saw one instance will be updated from t2.micro --> t2.medium. After this you went to grab a coffee without running terraform apply and

meanwhile a member of your team changed the instance type of that EC2 instance to t2.medium from aws console. After coming to your desk you run terraform apply. What will happen?

- A. 1 resource will be updated and you will see the message : Apply Complete !  
Resources : 0 added, 1 changed, 0 destroyed.
- B. No resource will be updated and you will see the message : Apply Complete !  
Resources : 0 added, 0 changed, 0 destroyed.
- C. terraform apply will through an error.
- D. The instance type will be changed to t2.micro and again will be changed to t2.medium

**Answer: B ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 94**

A variable az has the following default value. What will be the datatype of the variable?

az=["us-west-1a","us-east-1a"]

- A. Object
- B. String
- C. List
- D. Map

**Answer: C ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 95**

ABC Enterprise has recently tied up with multiple small organizations for exchanging database information.

Due to this, the firewall rules are increasing and are more than 100 rules. This is leading firewall configuration file that is difficult to manage. What is the way this type of configuration can be managed easily?

- A. Terraform Functions
- B. Terraform Backends
- C. Dynamic Blocks
- D. Terraform Expression

**Answer: ([SHOW ANSWER](#))**

#### **NEW QUESTION: 96**

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 plan 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.

**Answer: C ([LEAVE A REPLY](#))**

Reference:

<https://www.google.com/search?q=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new+infrastructure.&oq=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new+infrastructure.&aqs=chrome..69i57.556j0j7&sourceid=chrome&ie=UTF-8>

## NEW QUESTION: 97

When writing Terraform code, HashiCorp recommends that you use how many spaces between each nesting level?

- A. 0
- B. 1
- C. 2
- D. 4

**Answer: ([SHOW ANSWER](#))**

Explanation

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.

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:

```
ami = "abc123"  
instance_type = "t2.micro"
```

When both arguments and blocks appear together inside a block body, place all of the arguments together at the top and then place nested blocks below them. Use one blank line to separate the arguments from the blocks.

Use empty lines to separate logical groups of arguments within a block.

For blocks that contain both arguments and "meta-arguments" (as defined by the Terraform language semantics), list meta-arguments first and separate them from other arguments with one blank line. Place meta-argument blocks last and separate them from other blocks with one blank line.

```
resource "aws_instance" "example" {
```

```
count = 2 # meta-argument first
ami = "abc123"
instance_type = "t2.micro"
network_interface {
# ...
}
lifecycle { # meta-argument block last
create_before_destroy = true
}
}
```

Top-level blocks should always be separated from one another by one blank line. Nested blocks should also be separated by blank lines, except when grouping together related blocks of the same type (like multiple provisioner blocks in a resource).

Avoid separating multiple blocks of the same type with other blocks of a different type, unless the block types are defined by semantics to form a family. (For example: root\_block\_device, ebs\_block\_device and ephemeral\_block\_device on aws\_instance form a family of block types describing AWS block devices, and can therefore be grouped together and mixed.)

## NEW QUESTION: 98

In terraform, most resource dependencies are handled automatically. Which of the following statements describes best how terraform resource dependencies are handled?

- A.** Resource dependencies are identified and maintained in a file called resource.dependencies. Each terraform provider is required to maintain a list of all resource dependencies for the provider and it's included with the plugin during initialization when terraform init is executed. The file is located in the terraform.d folder.
- B.** The terraform binary contains a built-in reference map of all defined Terraform resource dependencies.
- C.** Resource dependencies are handled automatically by the depends\_on meta\_argument, which is set to true by default.
- D.** 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.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/resources.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!  
Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com  
**TA-002-P exam questions have been updated and answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

# HashiCorp.TA-002-P.v2020-10-08.q61

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	61
<b>Version:</b>	v2020-10-08
<b># of views:</b>	2164
<b># of Questions views:</b>	53118

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2020-10-08.q61.html>

## NEW QUESTION: 1

A colleague has informed you that a new version of a Terraform module that your team hosts on an Amazon S3 bucket is broken. The Amazon S3 bucket has versioning enabled. Your colleague tells you to make sure you are not using the latest version in your configuration. You have the following configuration block in your code that refers to the module:

module "infranet" { source = "s3::https://s3-us-west-2.amazonaws.com/infrabucket/infra\_module.zip"} What is the best way to ensure that you are not using the latest version of the module?

- A. Add a module version constraint in your configuration's backend block and specify a previous version.
- B. Add a version key to the module configuration and specify a previous version.
- C. Delete the latest version of the module in S3 to rollback to the previous version.
- D. Add a version property to the module in Terraform's state file and specify a previous version.

**Answer:** ([SHOW ANSWER](#))

Explanation

Version constraints are supported only for modules installed from a module registry, such as the 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.

Only Terraform Registries support module versioning by using the version key, one cannot configure a previous version of the module in the configuration. Deleting the latest version of the module in S3 is the only option of the available options that ensures you won't use the latest version. You could also modify the source URL to specify a versionId URL parameter for a previous version.

<https://www.terraform.io/docs/configuration/modules.html#source>

### **NEW QUESTION: 2**

What kind of resource dependency is stored in terraform.tfstate file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 3**

What does terraform refresh command do?

- A. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- B. terraform refresh is used to change/modify the infrastructure based on the existing state file, at that moment.
- C. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- D. terraform refresh syncs the state file with the real world infrastructure.
- E. terraform refresh command basically updates the configuration file with the current state of the actual infrastructure

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 4**

You have declared a variable name my\_var in terraform configuration without a value associated with it.

variable my\_var {}

After running terraform plan it will show an error as variable is not defined.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

Input variables are usually defined by stating a name, type and a default value. However, the type and default values are not strictly necessary. Terraform can deduct the type of the variable from the default or input value.

Variables can be predetermined in a file or included in the command-line options. As such, the simplest variable is just a name while the type and value are selected based on the input.

```
variable "variable_name" {}  
terraform apply -var variable_name="value"
```

The input variables, like the one above, use a couple of different types: strings, lists, maps, and boolean. Here are some examples of how each type are defined and used.

### String

Strings mark a single value per structure and are commonly used to simplify and make complicated values more user-friendly. Below is an example of a string variable definition.

```
variable "template" {  
  type = string  
  default = "01000000-0000-4000-8000-000030080200"  
}
```

A string variable can then be used in resource plans. Surrounded by double quotes, string variables are a simple substitution such as the example underneath.

```
storage = var.template
```

### List

Another type of Terraform variables lists. They work much like a numbered catalogue of values. Each value can be called by their corresponding index in the list. Here is an example of a list variable definition.

```
variable "users" {  
  type = list  
  default = ["root", "user1", "user2"]  
}
```

Lists can be used in the resource plans similarly to strings, but you'll also need to denote the index of the value you are looking for.

```
username = var.users[0]
```

### Map

Maps are a collection of string keys and string values. These can be useful for selecting values based on predefined parameters such as the server configuration by the monthly price.

```
variable "plans" {  
  type = map  
  default = {  
    "5USD" = "1xCPU-1GB"  
    "10USD" = "1xCPU-2GB"  
    "20USD" = "2xCPU-4GB"
```

```
}
```

```
}
```

You can access the right value by using the matching key. For example, the variable below would set the plan to "1xCPU-1GB".

```
plan = var.plans["5USD"]
```

The values matching to their keys can also be used to look up information in other maps.

For example, underneath is a shortlist of plans and their corresponding storage sizes.

```
variable "storage_sizes" {
```

```
  type = map
```

```
  default = {
```

```
    "1xCPU-1GB" = "25"
```

```
    "1xCPU-2GB" = "50"
```

```
    "2xCPU-4GB" = "80"
```

```
}
```

```
}
```

These can then be used to find the right storage size based on the monthly price as defined in the previous example.

```
size = lookup(var.storage_sizes, var.plans["5USD"])
```

Boolean

The last of the available variable type is boolean. They give the option to employ simple true or false values.

For example, you might wish to have a variable that decides when to generate the root user password on a new deployment.

```
variable "set_password" {
```

```
  default = false
```

```
}
```

The above example boolean can be used similarly to a string variable by simply marking down the correct variable.

```
create_password = var.set_password
```

By default, the value is set to false in this example. However, you can overwrite the variable at deployment by assigning a different value in a command-line variable.

```
terraform apply -var set_password="true"
```

## NEW QUESTION: 5

Complete the following sentence:

For local state, the workspaces are stored directly in a \_\_\_\_\_.

- A. a file called terraform.tfstate.backup
- B. directory called terraform.workspaces.tfstate
- C. a file called terraform.tfstate
- D. directory called terraform.tfstate.d

**Answer: D ([LEAVE A REPLY](#))**

## Explanation

For local state, Terraform stores the workspace states in a directory called `terraform.tfstate.d`.

<https://www.terraform.io/docs/state/workspaces.html#workspace-internals>

## NEW QUESTION: 6

Once a resource is marked as tainted, the next plan will show that the resource will be \_\_\_\_\_ and \_\_\_\_\_

\_\_\_\_\_ and the next apply will implement this change.

- A. destroyed and recreated
- B. destroyed and not recreated
- C. tainted and not destroyed
- D. recreated and tainted

**Answer:** A ([LEAVE A REPLY](#))

## NEW QUESTION: 7

You have created a terraform script that uses a lot of new constructs that have been introduced in terraform v0.12. However, many developers who are cloning the script from your git repo, are using v0.11, and getting errors. What can be done from your end to solve this problem?

- A. Force developer to use v0.12 by using terraform setting '`required_version`' and set it to `>=0.12`.
- B. Refactor the code to support both v0.11, and v0.12. It might be a difficult process, but there is no other way.
- C. Add a condition in front of each such specific construct, to check whether the running terraform version is v0.11 or v0.12, and work accordingly.
- D. Add comments in your code to tell developers to use v0.12 . If they use v0.11 , that should be their problem , which they need to figure out.

**Answer:** ([SHOW ANSWER](#))

## Explanation

<https://www.terraform.io/docs/configuration/terraform.html>

## NEW QUESTION: 8

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer:** A ([LEAVE A REPLY](#))

## Explanation

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

## NEW QUESTION: 9

What is a downside to using the Vault provider to read secrets from Vault?

- A. Secrets are persisted to the state file and plans.
- B. Terraform and Vault must be running on the same version.
- C. Terraform and Vault must be running on the same physical host.
- D. Terraform requires a unique auth method to work with Vault.

**Answer: A ([LEAVE A REPLY](#))**

## Explanation

The Vault provider allows Terraform to read from, write to, and configure Hashicorp Vault. Interacting with Vault from Terraform causes any secrets that you read and write to be persisted in both Terraform's state file and in any generated plan files. For any Terraform module that reads or writes Vault secrets, these files should be treated as sensitive and protected accordingly.

## NEW QUESTION: 10

A user has created three workspaces using the command line - prod, dev, and test. The user wants to create a fourth workspace named stage. Which command will the user execute to accomplish this?

- A. `terraform workspace new stage`
- B. `terraform workspace -new stage`
- C. `terraform workspace -create stage`
- D. `terraform workspace create stage`

**Answer: A ([LEAVE A REPLY](#))**

## Explanation

The `terraform workspace new` command is used to create a new workspace.

<https://www.terraform.io/docs/commands/workspace/new.html>

## NEW QUESTION: 11

Which of the following Terraform commands will automatically refresh the state unless supplied with additional flags or arguments? Choose TWO correct answers.

- A. `terraform validate`
- B. `terraform output`
- C. `terraform apply`
- D. `terraform plan`
- E. `terraform state`

**Answer: C,D** ([LEAVE A REPLY](#))

### **NEW QUESTION: 12**

Terraform must track metadata such as resource dependencies. Where is this data stored?

- A. workspace
- B. backend
- C. state file
- D. metadata store

**Answer: C** ([LEAVE A REPLY](#))

Explanation

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 13**

Which of the below datatype is not supported by Terraform.

- A. List
- B. Map
- C. Array
- D. Object

**Answer: C** ([LEAVE A REPLY](#))

### **NEW QUESTION: 14**

Which of the below backends support state locking?

- A. azurerm
- B. consul

C. S3

D. artifactory

**Answer: A,B,C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 15**

Command terraform refresh will update state file?

A. False

B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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. If the state is changed, this may cause changes to occur during the next plan or apply.

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

### **NEW QUESTION: 16**

You wanted to destroy some of the dependent resources from real infrastructure. You choose to delete those resources from your configuration file and run terraform plan and then apply. Which of the following way your resources would be destroyed?

A. Terraform can still determine the correct order for destruction from the state even when you delete one or more items from the configuration.

B. Those would be destroyed in the order in which they were written in the configuration file previously before you have deleted them from configuration file.

C. The resource will be destructed in random order as you have already deleted them from configuration.

D. You can not destroy resources by deleting them from configuration file and running plan and apply.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF** **Special Discount: freecram**)

### **NEW QUESTION: 17**

Which of the following clouds does not have a provider maintained HashiCorp?

- A. IBM Cloud
- B. DigitalOcean
- C. OpenStack
- D. AWS

**Answer: A ([LEAVE A REPLY](#))**

Explanation

IBM Cloud does not have a provider maintained by HashiCorp, although IBM Cloud does maintain their own Terraform provider.

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

### **NEW QUESTION: 18**

Your organization has moved to AWS and has manually deployed infrastructure using the console. Recently, a decision has been made to standardize on Terraform for all deployments moving forward.

What can you do to ensure that all existing is managed by Terraform moving forward without interruption to existing services?

- A. Submit a ticket to AWS and ask them to export the state of all existing resources and use terraform import to import them into the state file.
- B. Delete the existing resources and recreate them using new a Terraform configuration so Terraform can manage them moving forward.
- C. Resources that are manually deployed in the AWS console cannot be imported by Terraform.
- D. Using terraform import, import the existing infrastructure into your Terraform state.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The `terraform import` command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

## NEW QUESTION: 19

Which Terraform command will check and report errors within modules, attribute names, and value types to make sure they are syntactically valid and internally consistent?

- A. `terraform validate`
- B. `terraform format`
- C. `terraform fmt`
- D. `terraform show`

**Answer: ([SHOW ANSWER](#))**

Explanation

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 the 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.

## NEW QUESTION: 20

Named workspaces are not a suitable isolation mechanism for strong separation between staging and production?

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

## NEW QUESTION: 21

What resource dependency information is stored in Terraform's state?

- A. Only implicit dependencies are stored in state.
- B. Both implicit and explicit dependencies are stored in state.
- C. Only explicit dependencies are stored in state.
- D. No dependency information is stored in state.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

## NEW QUESTION: 22

Which of the following actions are performed during a terraform init?

- A. Initializes downloaded and/or installed providers
- B. Initializes the backend configuration
- C. Provisions the declared resources in your configuration
- D. Download the declared providers which are supported by HashiCorp

**Answer: ([SHOW ANSWER](#))**

Explanation

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

This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times.

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.

terraform init command does -

- \* Copy a Source Module
- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

### **NEW QUESTION: 23**

Which of the below are paid features of Terraform Cloud?

- A.** Full API Coverage
- B.** Secure variable Storage
- C.** Roles/ Team management
- D.** Cost Estimation
- E.** Private Module Registry
- F.** Sentinel policies

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 24**

When multiple engineers start deploying infrastructure using the same state file, what is a feature of remote state storage that is critical to ensure the state doesn't become corrupt?

- A.** Object Storage
- B.** State Locking
- C.** WorkSpaces
- D.** Encryption

**Answer:** **B** ([LEAVE A REPLY](#))

Explanation

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.

State locking happens automatically on all operations that could write state. You won't see any message that it is happening. If state locking fails, Terraform will not continue. You can disable state locking for most commands with the `-lock` flag but it is not recommended.

If acquiring the lock is taking longer than expected, Terraform will output a status message. If Terraform doesn't output a message, state locking is still occurring if your backend supports it.

Not all backends support locking. Please view the list of backend types for details on whether a backend supports locking or not.

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

### **NEW QUESTION: 25**

The Security Operations team of ABC Enterprise wants to mandate that all the Terraform configuration that creates an S3 bucket must have encryption feature enabled. What is the best way to achieve it?

- A.** Use Sentinel Policies.
- B.** Use S3 bucket policy.
- C.** Create a script that checks the encryption parameter is enabled on every git commit.
- D.** Shared a SOP to engineers to mandate encryption feature on S3.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

Using Sentinel with Terraform Cloud involves:

- \* Defining the policies - Policies are defined using the policy language with imports for parsing the Terraform plan, state and configuration.
- \* Managing policies for organizations - Users with permission to manage policies can add policies to their organization by configuring VCS integration or uploading policy sets through the API. They also define which workspaces the policy sets are checked against during runs. (More about permissions.)
- \* 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.
- \* Mocking Sentinel Terraform data - Terraform Cloud provides the ability to generate mock data for any run within a workspace. This data can be used with the Sentinel CLI to test policies before deployment.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### **NEW QUESTION: 26**

lookup retrieves the value of a single element from which of the below data type?

- A.** map
- B.** set
- C.** string
- D.** list

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

## **NEW QUESTION: 27**

In regards to deploying resources in multi-cloud environments, what are some of the benefits of using Terraform rather than a provider's native tooling? (select three)

- A.** Terraform can help businesses deploy applications on multiple clouds and on-premises infrastructure.
- B.** Terraform is not cloud-agnostic and can be used to deploy resources across a single public cloud.
- C.** Terraform simplifies management and orchestration, helping operators build large-scale, multi-cloud infrastructure.
- D.** Terraform can manage cross-cloud dependencies.

**Answer: A,C,D ([LEAVE A REPLY](#))**

Explanation

Terraform is cloud-agnostic and allows a single configuration to be used to manage multiple providers, and to even handle cross-cloud dependencies. This simplifies management and orchestration, helping operators build large-scale multi-cloud infrastructures.

<https://www.terraform.io/intro/use-cases.html>

## **NEW QUESTION: 28**

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A.** TF\_TRACE
- B.** TF\_DEBUG
- C.** TF\_LOG
- D.** TF\_INFO

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

## **NEW QUESTION: 29**

What is the command you can use to set an environment variable named "var1" of type String?

- A.** export TF\_VAR\_VAR1
- B.** set TF\_VAR\_var1
- C.** variable "var1" { type = "string"}

D. export TF\_VAR\_var1

**Answer: D ([LEAVE A REPLY](#))**

Explanation

The environment variable must be in the format TF\_VAR\_name, so for the question TF\_VAR\_var1 is the correct choice.

[https://www.terraform.io/docs/commands/environment-variables.html#tf\\_var\\_name](https://www.terraform.io/docs/commands/environment-variables.html#tf_var_name)

### **NEW QUESTION: 30**

A variable az has the following default value. What will be the datatype of the variable?

az=["us-west-1a","us-east-1a"]

- A. Object
- B. Map
- C. String
- D. List

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 31**

In regards to Terraform state file, select all the statements below which are correct?

- A. When using local state, the state file is stored in plain-text.
- B. The state file is always encrypted at rest.
- C. Storing state remotely can provide better security.
- D. Using the mask feature, you can instruct Terraform to mask sensitive data in the state file.
- E. The Terraform state can contain sensitive data, therefore the state file should be protected from unauthorized access.
- F. Terraform Cloud always encrypts state at rest.

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform state can contain sensitive data, depending on the resources in use and your definition of

"sensitive." The state contains resource IDs and all resource attributes. For resources such as databases, this may contain initial passwords.

When using local state, state is stored in plain-text JSON files.

When using remote state, state is only ever held in memory when used by Terraform. It may be encrypted at rest, but this depends on the specific remote state backend.

Storing Terraform 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.

Recommendations

If you manage any sensitive data with Terraform (like database passwords, user passwords, or private keys), treat the state itself as sensitive data.

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.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 32

You cannot publish your own modules on the Terraform Registry.

A. False

B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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

## NEW QUESTION: 33

Valarie has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code. Valarie wants to hide the output value in the CLI after terraform apply that's why she has used sensitive parameter.

1. output "db\_password" {
2. value = local.db\_password
3. sensitive = true
4. }

Since sensitive is set to true, will the value associated with db password be available in plain-text in the state file for everyone to read?

A. Yes

B. No

## **Answer: A ([LEAVE A REPLY](#))**

### Explanation

Outputs can be marked as containing sensitive material by setting the sensitive attribute to true, like this:

```
output "sensitive" {  
    sensitive = true  
    value = VALUE  
}
```

When outputs are displayed on-screen following a terraform apply or terraform refresh, sensitive outputs are redacted, with <sensitive> displayed in place of their value.

### Limitations of Sensitive Outputs

The values of sensitive outputs are still stored in the Terraform state, and available using the terraform output command, so cannot be relied on as a sole means of protecting values.

Sensitivity is not tracked internally, so if the output is interpolated in another module into a resource, the value will be displayed.

## **NEW QUESTION: 34**

Mary has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code:

1. output "db\_password"
2. {
3. value = local.db\_password
4. }

Mary wants to hide the output value in the CLI after terraform apply? What is the best way?

- A. Use cryptographic hash
- B. Use sensitive parameter
- C. Encrypt the value using encrypt() function
- D. Use secure parameter

## **Answer: B ([LEAVE A REPLY](#))**

## **NEW QUESTION: 35**

Which of the below options is a valid interpolation syntax for retrieving a data source?

- A. \${google\_storage\_bucket.backend}
- B. \${azurerm\_resource\_group.test.data}
- C. \${aws\_instance.web.id.data}
- D. \${data.google\_dns\_keys.foo\_dns\_keys.key\_signing\_keys[0].ds\_record}

## **Answer: D ([LEAVE A REPLY](#))**

### Explanation

Data source attributes are interpolated with the general syntax `data.TYPE.NAME.ATTRIBUTE`. The interpolation for a resource is the same but without the `data.` prefix (`TYPE.NAME.ATTRIBUTE`).

<https://www.terraform.io/docs/configuration-0-11/interpolation.html#attributes-of-a-data-source>

## NEW QUESTION: 36

Given the Terraform configuration below, in which order will the resources be created?

```
1. resource "aws_instance" "web_server"
2. {
3.   ami = "ami-b374d5a5"
4.   instance_type = "t2.micro"
5. }
6. resource "aws_eip" "web_server_ip"
7. {
8.   vpc = true
9.   instance = aws_instance.web_server.id
}
```

- A. `aws_eip` will be created first  
`aws_instance` will be created second
- B. `aws_eip` will be created first  
`aws_instance` will be created second
- C. Resources will be created simultaneously
- D. `aws_instance` will be created first  
`aws_eip` will be created second

**Answer: D (LEAVE A REPLY)**

Explanation

Implicit and Explicit Dependencies

By studying the resource attributes used in interpolation expressions, Terraform can automatically infer when one resource depends on another. In the example above, the reference to `aws_instance.web_server.id` creates an implicit dependency on the `aws_instance` named `web_server`.

Terraform uses this dependency information to determine the correct order in which to create the different resources.

```
# Example of Implicit Dependency
resource "aws_instance" "web_server" {
  ami = "ami-b374d5a5"
  instance_type = "t2.micro"
}
resource "aws_eip" "web_server_ip" {
  vpc = true
  instance = aws_instance.web_server.id
}
```

```
}
```

In the example above, Terraform knows that the `aws_instance` must be created before the `aws_eip`.

Implicit dependencies via interpolation expressions are the primary way to inform Terraform about these relationships, and should be used whenever possible.

Sometimes there are dependencies between resources that are not visible to Terraform. The `depends_on` argument is accepted by any resource and accepts a list of resources to create explicit dependencies for.

For example, perhaps an application we will run on our EC2 instance expects to use a specific Amazon S3 bucket, but that dependency is configured inside the application code and thus not visible to Terraform. In that case, we can use `depends_on` to explicitly declare the dependency:

```
# Example of Explicit Dependency
```

```
# New resource for the S3 bucket our application will use.
```

```
resource "aws_s3_bucket" "example" {  
  bucket = "terraform-getting-started-guide"  
  acl = "private"  
}
```

```
# Change the aws_instance we declared earlier to now include "depends_on" resource  
"aws_instance" "example" { ami = "ami-2757f631" instance_type = "t2.micro"
```

```
# Tells Terraform that this EC2 instance must be created only after the  
# S3 bucket has been created.
```

```
depends_on = [aws_s3_bucket.example]  
}
```

<https://learn.hashicorp.com/terraform/getting-started/dependencies.html>

## NEW QUESTION: 37

What does terraform plan do ?

- A. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.
- B. Create an execution plan by evaluating the difference between configuration file and state file.
- C. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.
- D. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.

**Answer: B ([LEAVE A REPLY](#))**

## NEW QUESTION: 38

Terraform import command can import resources into modules as well directly into the root of your state.

**A.** True

**B.** False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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 directly into the root of your state.

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform. For example:

```
resource "aws_instance" "import_example" {  
  # ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration:

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

As a result of the above command, the resource is recorded in the state file. We can now run terraform plan to see how the configuration compares to the imported resource, and make any adjustments to the configuration to align with the current (or desired) state of the imported object.

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

## NEW QUESTION: 39

You are using a terraform operation that writes state. Unfortunately automatic state unlocking has failed for that operation. Which of the below commands can be used to remove the already acquired lock on the state?

- A. terraform unlock
- B. terraform force-unlock
- C. terraform state unlock
- D. None of the above

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Command: force-unlock

Manually unlock the state for the defined configuration.

This will not modify your infrastructure. This command removes the lock on the state for the current configuration. The behavior of this lock is dependent on the backend being used. Local state files cannot be unlocked by another process.

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

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

Terraform has a force-unlock command to manually unlock the state if unlocking failed. 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.

#### **NEW QUESTION: 40**

Terraform works well in Windows but a Windows server is required.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

You may see this question in actual exam. Please remember : Terraform does not require GO language to be installed as a prerequisite and it does not require a Windows Server as well.

#### **NEW QUESTION: 41**

You have created a custom variable definition file my\_vars.tfvars. How will you use it for provisioning infrastructure?

- A. terraform apply -var-state-file ="my\_vars.tfvars"
- B. terraform apply var-file="my\_vars.tfvars"
- C. terraform plan -var-file="my\_vars.tfvar"
- D. terraform apply -var-file="my\_vars.tfvars"

**Answer: D ([LEAVE A REPLY](#))**

Explanation

To set lots of variables, it is more convenient to specify their values in a variable definitions file (with a filename ending in either .tfvars or .tfvars.json) and then specify that file on the command line with -var-file:

`terraform apply -var-file="my_vars.tfvars"`

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

### **NEW QUESTION: 42**

Your company has a lot of workloads in AWS , and Azure that were respectively created using CloudFormation , and AzureRM Templates. However , now your CIO has decided to use Terraform for all new projects , and has asked you to check how to integrate the existing environment with terraform code.

What should be your next plan of action?

- A. This is only possible in Terraform Enterprise , which has the TerraformConverter exe that can take any other template language like AzureRM and convert to Terraform code.
- B. Tell the CIO that this is not possible . Resources created in CloudFormation , and AzureRM templates cannot be tracked using terraform.
- C. Just write the terraform config file for the new resources , and run terraform apply , the state file will automatically be updated with the details of the new resources to be imported.
- D. Use terraform import command to import each resource one by one .

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 43**

In Terraform Enterprise, a workspace can be mapped to how many VCS repos?

- A. 5
- B. 2
- C. 3
- D. 1

**Answer: D ([LEAVE A REPLY](#))**

Explanation

A workspace can only be configured to a single VCS repo, however, multiple workspaces can use the same repo.

<https://www.terraform.io/docs/cloud/workspaces/vcs.html>

### **NEW QUESTION: 44**

1. resource "aws\_s3\_bucket" "example" {
2. bucket = "my-test-s3-terraform-bucket"
3. ...} resource "aws\_iam\_role" "test\_role" {
4. name = "test\_role"
5. ...}

Due to the way that the application code is written, the s3 bucket must be created before the test role is created, otherwise there will be a problem. How can you ensure that?

- A. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- B. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.

- C. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- D. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Implicit dependency works only if there is some reference of one resource to another.  
Explicit dependency is the option here.

### **NEW QUESTION: 45**

You want to use terraform import to start managing infrastructure that was not originally provisioned through infrastructure as code. Before you can import the resource's current state, what must you do in order to prepare to manage these resources using Terraform?

- A. Run terraform refresh to ensure that the state file has the latest information for existing resources.
- B. Update the configuration file to include the new resources.
- C. Shut down or stop using the resources being imported so no changes are inadvertently missed.
- D. Modify the Terraform state file to add the new resources.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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. The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import  
successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

### NEW QUESTION: 46

By default, a defined provisioner is a creation-time provisioner.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

### NEW QUESTION: 47

Which of the following allows Terraform users to apply policy as code to enforce standardized configurations for resources being deployed via infrastructure as code?

- A. Sentinel
- B. Module registry
- C. Functions
- D. Workspaces

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Sentinel is a language and framework for policy built to be embedded in existing software to enable fine-grained, logic-based policy decisions. A policy describes under what circumstances certain behaviors are allowed. Sentinel is an enterprise-only feature.

[https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb\\_title](https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb_title)

### NEW QUESTION: 48

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

- A. False

**B.** True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

You can change your backend configuration at any time. You can change both the configuration itself as well as the type of backend (for example from "consul" to "s3"). Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

### **NEW QUESTION: 49**

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

- A.** terraform providers
- B.** terraform state show
- C.** terraform provider
- D.** terraform plan

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Use the `terraform providers` command to view the specified version constraints for all providers used in the current configuration.

<https://www.terraform.io/docs/configuration/providers.html>

### **NEW QUESTION: 50**

Multiple provider instances blocks for AWS can be part of a single configuration file?

- A.** False
- B.** True

**Answer: ([SHOW ANSWER](#))**

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}
```

```
# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it creates an additional provider configuration. For providers that have no required configuration arguments, the implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-provider-instances>

### **NEW QUESTION: 51**

Multiple providers can be declared within a single Terraform configuration file.

A. True

B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration.

For Example

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it creates an additional provider configuration. For providers that have no required configuration arguments, the implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html>

## **NEW QUESTION: 52**

Terraform-specific settings and behaviors are declared in which configuration block type?

- A. provider
- B. terraform
- C. resource
- D. data

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The special terraform configuration block type is used to configure some behaviors of Terraform itself, such as requiring a minimum Terraform version to apply your configuration.

Example

```
terraform {  
  required_version = "> 0.12.0"  
}
```

<https://www.terraform.io/docs/configuration/terraform.html>

## **NEW QUESTION: 53**

Which of the following challenges would Terraform be a candidate for solving? (Select THREE)

- A. Reduce the number of workflows needed for managing infrastructure across each of the companies public and private clouds.
- B. Utilize a single tool for all of the infrastructure and configuration management needs.
- C. Enable self-service infrastructure to allocate resources on your proprietary private cloud.
- D. Have a single interoperable tool to manage the variety of services including GitHub repositories, MySQL database, and Kubernetes clusters.

**Answer: ([SHOW ANSWER](#))**

## **NEW QUESTION: 54**

Using multi-cloud and provider-agnostic tools provides which of the following benefits?

- A. Operations teams only need to learn and manage a single tool to manage infrastructure, regardless of where the infrastructure is deployed.
- B. Increased risk due to all infrastructure relying on a single tool for management.
- C. Can be used across major cloud providers and VM hypervisors.
- D. Slower provisioning speed allows the operations team to catch mistakes before they are applied.

**Answer: A,C ([LEAVE A REPLY](#))**

Explanation

Using a tool like Terraform can be advantageous for organizations deploying workloads across multiple public and private cloud environments. Operations teams only need to

learn a single tool, single language, and can use the same tooling to enable a DevOps-like experience and workflows.

### **NEW QUESTION: 55**

From the answers below, select the advantages of using Infrastructure as Code.

- A.** Provide a codified workflow to develop customer-facing applications.
- B.** Safely test modifications using a "dry run" before applying any actual changes.
- C.** Easily integrate with application workflows (GitLab Actions, Azure DevOps, CI/CD tools).
- D.** Easily change and update existing infrastructure.
- E.** Provide reusable modules for easy sharing and collaboration.

**Answer:** ([SHOW ANSWER](#))

Explanation

Infrastructure as Code is not used to develop applications, but it can be used to help deploy or provision those applications to a public cloud provider or on-premises infrastructure.

All of the others are benefits to using Infrastructure as Code over the traditional way of managing infrastructure, regardless if it's public cloud or on-premises.

### **NEW QUESTION: 56**

What is the default backend for Terraform?

- A.** consul
- B.** gcs
- C.** local
- D.** etcd

**Answer:** C ([LEAVE A REPLY](#))

Explanation

By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to.

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

### **NEW QUESTION: 57**

Which of the following is the right substitute for static values that can make Terraform configuration file more dynamic and reusable?

- A.** Output value
- B.** Input parameters
- C.** Functions
- D.** Modules

**Answer:** ([SHOW ANSWER](#))

Explanation

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module's own source code, and allowing modules to be shared between different configurations.

### **NEW QUESTION: 58**

You have already set TF\_LOG = DEBUG to enable debug log. Now you want to always write the log to the directory you're currently running terraform from. what should you do to achieve this.

- A. Run the command export TF\_LOG\_FILE=./terraform.log.
- B. Run the command export TF\_LOG\_PATH=./terraform.log.
- C. Run the command export TF\_DEBUG\_PATH=./terraform.log.
- D. No explicit action required. Terraform will take care of this as you have enable TF\_LOG.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/commands/environment-variables.html>

### **NEW QUESTION: 59**

You have created a custom variable definition file testing.tfvars. How will you use it for provisioning infrastructure?

- A. terraform apply -var-state-file ="testing.tfvars"
- B. terraform plan -var-file="testing.tfvar"
- C. terraform apply -var-file="testing.tfvars"
- D. terraform apply var-file="testing.tfvars"

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 60**

Which of the following state management command allow you to retrieve a list of resources that are part of the state file?

- A. terraform state list
- B. terraform state view
- C. terraform view
- D. terraform list

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The terraform state list command is used to list resources within a Terraform state.

Usage: terraform state list [options] [address...]

The command will list all resources in the state file matching the given addresses (if any). If no addresses are given, all resources are listed.

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

## NEW QUESTION: 61

State locking does not happen automatically and must be specified at run

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

State locking happens automatically on all operations that could write state.

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

# HashiCorp.TA-002-P.v2021-12-18.q113

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	113
<b>Version:</b>	v2021-12-18
<b># of views:</b>	149
<b># of Questions views:</b>	2409

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2021-12-18.q113.html>

## NEW QUESTION: 1

If a module uses a local variable, you can expose that value with a terraform output.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Output values are like function return values.

## NEW QUESTION: 2

When writing Terraform code, HashiCorp recommends that you use how many spaces between each nesting level?

- A. 0
- B. 1
- C. 2
- D. 4

**Answer: C ([LEAVE A REPLY](#))**

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.

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:

```
ami = "abc123"
```

```
instance_type = "t2.micro"
```

When both arguments and blocks appear together inside a block body, place all of the arguments together at the top and then place nested blocks below them. Use one blank line to separate the arguments from the blocks.

Use empty lines to separate logical groups of arguments within a block.

For blocks that contain both arguments and "meta-arguments" (as defined by the Terraform language semantics), list meta-arguments first and separate them from other arguments with one blank line. Place meta-argument blocks last and separate them from other blocks with one blank line.

```
resource "aws_instance" "example" {
  count = 2 # meta-argument first
  ami   = "abc123"
  instance_type = "t2.micro"
  network_interface {
    # ...
  }
  lifecycle { # meta-argument block last
    create_before_destroy = true
  }
}
```

Top-level blocks should always be separated from one another by one blank line. Nested blocks should also be separated by blank lines, except when grouping together related blocks of the same type (like multiple provisioner blocks in a resource).

Avoid separating multiple blocks of the same type with other blocks of a different type, unless the block types are defined by semantics to form a family. (For example: root\_block\_device, ebs\_block\_device and ephemeral\_block\_device on aws\_instance form a family of block types describing AWS block devices, and can therefore be grouped together and mixed.)

### NEW QUESTION: 3

What is the result of the following terraform function call?

- A. hello
- B. what?
- C. goodbye

**Answer: B ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

### NEW QUESTION: 4

Which of the below configuration file formats are supported by Terraform? (Select TWO)

- A. Node
- B. JSON

- C. Go
- D. YAML
- E. HCL

**Answer:** ([SHOW ANSWER](#))

Terraform supports both HashiCorp Configuration Language (HCL) and JSON formats for configurations.

<https://www.terraform.io/docs/configuration/>

### **NEW QUESTION: 5**

True or False? When using the Terraform provider for Vault, the tight integration between these HashiCorp tools provides the ability to mask secrets in the terraform plan and state files.

- A. False
- B. True

**Answer:** A ([LEAVE A REPLY](#))

Explanation

Currently, Terraform has no mechanism to redact or protect secrets that are returned via data sources, so secrets read via this provider will be persisted into the Terraform state, into any plan files, and in some cases in the console output produced while planning and applying. These artifacts must, therefore, all be protected accordingly.

### **NEW QUESTION: 6**

Terraform-specific settings and behaviors are declared in which configuration block type?

- A. provider
- B. terraform
- C. resource
- D. data

**Answer:** B ([LEAVE A REPLY](#))

The special terraform configuration block type is used to configure some behaviors of Terraform itself, such as requiring a minimum Terraform version to apply your configuration.

Example

```
terraform {  
  required_version = "> 0.12.0"  
}
```

<https://www.terraform.io/docs/configuration/terraform.html>

### **NEW QUESTION: 7**

What is one disadvantage of using dynamic blocks in Terraform?

- A. They cannot be used to loop through a list of values
- B. They make configuration harder to read and understand

- C. Dynamic blocks can construct repeatable nested blocks
- D. Terraform will run more slowly

**Answer:** A ([LEAVE A REPLY](#))

### **NEW QUESTION: 8**

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.

**Answer:**

Terraformtfstate

### **NEW QUESTION: 9**

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

**Answer:** C ([LEAVE A REPLY](#))

terraform workspace list

The command will list all existing workspaces.

### **NEW QUESTION: 10**

Anyone can publish and share modules on the Terraform Public Module Registry, and meeting the requirements for publishing a module is extremely easy. Select from the following list all valid requirements. (select three)

- A. The module must be PCI/HIPPA compliant.
- B. Module repositories must use this three-part name format, terraform-- .
- C. The registry uses tags to identify module versions.
- D. Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .
- E. The module must be on GitHub and must be a public repo.

**Answer:** C,D,E ([LEAVE A REPLY](#))

Explanation

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

### **NEW QUESTION: 11**

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.

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

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.

<https://www.terraform.io/docs/state/sensitive-data.html#recommendations>

**NEW QUESTION: 12**

In Terraform Enterprise, a workspace can be mapped to how many VCS repos?

- A. 5
- B. 2
- C. 3
- D. 1

**Answer: D ([LEAVE A REPLY](#))**

A workspace can only be configured to a single VCS repo, however, multiple workspaces can use the same repo.

<https://www.terraform.io/docs/cloud/workspaces/vcs.html>

**NEW QUESTION: 13**

True or False: Workspaces provide identical functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/cloud/workspaces/index.html>

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

**NEW QUESTION: 14**

What feature of Terraform Cloud and/or Terraform Enterprise can you publish and maintain a set of custom modules which can be used within your organization?

- A. custom VCS integration
- B. Terraform registry
- C. remote runs
- D. private module registry

**Answer: D ([LEAVE A REPLY](#))****NEW QUESTION: 15**

What does the default "local" Terraform backend store?

- A. tfplan files
- B. Terraform binary

C. Provider plugins

D. State file

**Answer: D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 16**

When Terraform needs to be installed in a location where it does not have internet access to download the installer and upgrades, the installation is generally known as to be \_\_\_\_\_.

A. a private install

B. disconnected

C. air-gapped

D. non-traditional

**Answer: ([SHOW ANSWER](#))**

Explanation

A Terraform Enterprise install that is provisioned on a network that does not have Internet access is generally known as an air-gapped install. These types of installs require you to pull updates, providers, etc. from external sources vs. being able to download them directly.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: **freecram****

### **NEW QUESTION: 17**

Terraform Cloud is more powerful when you integrate it with your version control system (VCS) provider. Select all the supported VCS providers from the answers below. (select four)

A. GitHub

B. CVS Version Control

C. Azure DevOps Server

D. Bitbucket Cloud

E. GitHub Enterprise

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform Cloud supports the following VCS providers:

- <https://www.terraform.io/docs/cloud/vcs/github.html>
  - <https://www.terraform.io/docs/cloud/vcs/github.html>
  - <https://www.terraform.io/docs/cloud/vcs/github-enterprise.html>
  - <https://www.terraform.io/docs/cloud/vcs/gitlab-com.html>
  - <https://www.terraform.io/docs/cloud/vcs/gitlab-eece.html>
  - <https://www.terraform.io/docs/cloud/vcs/bitbucket-cloud.html>
  - <https://www.terraform.io/docs/cloud/vcs/bitbucket-server.html>
  - <https://www.terraform.io/docs/cloud/vcs/azure-devops-server.html>
  - <https://www.terraform.io/docs/cloud/vcs/azure-devops-services.html>
- <https://www.terraform.io/docs/cloud/vcs/index.html#supported-vcs-providers>

### **NEW QUESTION: 18**

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 approve plans in VCS connected workspaces
- B.** Commit a change to the VCS working directory and branch that the Terraform Cloud workspace is connected to
- C.** Only Terraform Cloud organization owners can set workspace variables on VCS connected workspaces
- D.** Only members of a VCS organization can open a pull request against repositories that are connected to Terraform Cloud workspaces

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 19**

Which of the following actions are performed during a terraform init?

- A.** Initializes downloaded and/or installed providers
- B.** Initializes the backend configuration
- C.** Provisions the declared resources in your configuration
- D.** Download the declared providers which are supported by HashiCorp

**Answer: A,B,D ([LEAVE A REPLY](#))**

The terraform init command is used to initialize a working directory containing Terraform configuration files. This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times.

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.

terrafrom init command does -

- \* Copy a Source Module

- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

## NEW QUESTION: 20

Given the Terraform configuration below, in which order will the resources be created?

```
1. resource "aws_instance" "web_server"  
2. {  
3.   ami = "ami-b374d5a5"  
4.   instance_type = "t2.micro"  
5. }  
6. resource "aws_eip" "web_server_ip"  
7. {  
8.   vpc = true  
9.   instance = aws_instance.web_server.id  
 }
```

- A. aws\_eip will be created first  
aws\_instance will be created second
- B. aws\_eip will be created first  
aws\_instance will be created second
- C. Resources will be created simultaneously
- D. aws\_instance will be created first  
aws\_eip will be created second

### Answer: D ([LEAVE A REPLY](#))

Implicit and Explicit Dependencies

By studying the resource attributes used in interpolation expressions, Terraform can automatically infer when one resource depends on another. In the example above, the reference to `aws_instance.web_server.id` creates an implicit dependency on the `aws_instance` named `web_server`.

Terraform uses this dependency information to determine the correct order in which to create the different resources.

```
# Example of Implicit Dependency  
resource "aws_instance" "web_server" {  
  ami = "ami-b374d5a5"  
  instance_type = "t2.micro"  
}  
resource "aws_eip" "web_server_ip" {  
  vpc = true  
  instance = aws_instance.web_server.id  
}
```

In the example above, Terraform knows that the `aws_instance` must be created before the `aws_eip`.

Implicit dependencies via interpolation expressions are the primary way to inform Terraform about these relationships, and should be used whenever possible.

Sometimes there are dependencies between resources that are not visible to Terraform.

The `depends_on` argument is accepted by any resource and accepts a list of resources to create explicit dependencies for.

For example, perhaps an application we will run on our EC2 instance expects to use a specific Amazon S3 bucket, but that dependency is configured inside the application code and thus not visible to Terraform. In that case, we can use `depends_on` to explicitly declare the dependency:

```
# Example of Explicit Dependency
# New resource for the S3 bucket our application will use.
resource "aws_s3_bucket" "example" {
  bucket = "terraform-getting-started-guide"
  acl = "private"
}
# Change the aws_instance we declared earlier to now include "depends_on" resource
"aws_instance" "example" { ami = "ami-2757f631" instance_type = "t2.micro"
# Tells Terraform that this EC2 instance must be created only after the
# S3 bucket has been created.
depends_on = [aws_s3_bucket.example]
}
```

<https://learn.hashicorp.com/terraform/getting-started/dependencies.html>

## NEW QUESTION: 21

In order to reduce the time it takes to provision resources, Terraform uses parallelism. By default, how many resources will Terraform provision concurrently?

- A. 10
- B. 5
- C. 50
- D. 20

**Answer: A ([LEAVE A REPLY](#))**

## NEW QUESTION: 22

Every region in AWS has a different AMI ID for Linux and these are keep on changing.

What is the best approach to create the EC2 instances that can deal with different AMI IDs based on regions?

- A. Use data source `aws_ami`.
- B. Create a map of region to ami id.
- C. Create different configuration file for different region.

D. None of the above

**Answer: A ([LEAVE A REPLY](#))**

<https://www.terraform.io/docs/configuration/data-sources.html>

### **NEW QUESTION: 23**

Which backend does the Terraform CLI use by default?

- A. Consul
- B. Terraform Cloud
- C. Local
- D. Remote

**Answer: C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 24**

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A. Explicit use of a provider block in configuration, optionally including a version constraint.
- B. Use of any resource belonging to a particular provider in a resource or data block in configuration.
- C. Existence of any resource instance belonging to a particular provider in the current state.
- D. Existence of any provider plugins found locally in the working directory.

**Answer: A,B,C ([LEAVE A REPLY](#))**

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider dependency. The plugin can exist without any reference to it in the terraform configuration. <https://www.terraform.io/docs/commands/providers.html>

### **NEW QUESTION: 25**

You have provisioned some aws resources in your test environment through Terraform for a POC work. After the POC, now you want to destroy the resources but before destroying them you want to check what resources will be getting destroyed through terraform. what are the options of doing that? (Select TWO)

- A. Use terraform destroy command
- B. This is not possible
- C. Use terraform plan command
- D. Use terraform plan -destroy command.

**Answer: A,D ([LEAVE A REPLY](#))**

<https://learn.hashicorp.com/terraform/getting-started/destroy>

### **NEW QUESTION: 26**

Please identify the offerings which are unique to Terraform Enterprise, and not available in either Terraform OSS, or Terraform Cloud. Select four.

- A. Audit Logs
- B. Private Network Connectivity
- C. VCS Integration
- D. Sentinel
- E. Clustering

**Answer:** ([SHOW ANSWER](#))

<https://www.hashicorp.com/products/terraform/pricing/>

## NEW QUESTION: 27

State is a requirement for Terraform to function

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

State is a necessary requirement for Terraform to function. It is often asked if it is possible for Terraform to work without state, or for Terraform to not use state and just inspect cloud resources on every run.

Purpose of Terraform State

State is a necessary requirement for Terraform to function. It is often asked if it is possible for Terraform to work without state, or for Terraform to not use state and just inspect cloud resources on every run. This page will help explain why Terraform state is required.

As you'll see from the reasons below, state is required. And in the scenarios where Terraform may be able to get away without state, doing so would require shifting massive amounts of complexity from one place (state) to another place (the replacement concept).

### 1. Mapping to the Real World

Terraform requires some sort of database to map Terraform config to the real world. When you have a resource resource "aws\_instance" "foo" in your configuration, Terraform uses this map to know that instance i- abcd1234 is represented by that resource.

For some providers like AWS, Terraform could theoretically use something like AWS tags. Early prototypes of Terraform actually had no state files and used this method. However, we quickly ran into problems. The first major issue was a simple one: not all resources support tags, and not all cloud providers support tags.

Therefore, for mapping configuration to resources in the real world, Terraform uses its own state structure.

### 2. Metadata

Alongside the mappings between resources and remote objects, Terraform must also track metadata such as resource dependencies.

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your

configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state. Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

One way to avoid this would be for Terraform to know a required ordering between resource types. For example, Terraform could know that servers must be deleted before the subnets they are a part of. The complexity for this approach quickly explodes, however: in addition to Terraform having to understand the ordering semantics of every resource for every cloud, Terraform must also understand the ordering across providers.

Terraform also stores other metadata for similar reasons, such as a pointer to the provider configuration that was most recently used with the resource in situations where multiple aliased providers are present.

### 3. Performance

In addition to basic mapping, Terraform stores a cache of the attribute values for all resources in the state. This is the most optional feature of Terraform state and is done only as a performance improvement.

When running a terraform plan, Terraform must know the current state of resources in order to effectively determine the changes that it needs to make to reach your desired configuration.

For small infrastructures, Terraform can query your providers and sync the latest attributes from all your resources. This is the default behavior of Terraform: for every plan and apply, Terraform will sync all resources in your state.

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to query multiple resources at once, and the round trip time for each resource is hundreds of milliseconds. On top of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

### 4. Syncing

In the default configuration, Terraform stores the state in a file in the current working directory where Terraform was run. This is okay for getting started, but when using Terraform in a team it is important for everyone to be working with the same state so that operations will be applied to the same remote objects.

Remote state is the recommended solution to this problem. With a fully-featured state backend, Terraform can use remote locking as a measure to avoid two or more different users accidentally running Terraform at the same time, and thus ensure that each Terraform run begins with the most recent updated state.

What resource dependency information is stored in Terraform's state?

- A. Only implicit dependencies are stored in state.
- B. Both implicit and explicit dependencies are stored in state.
- C. Only explicit dependencies are stored in state.
- D. No dependency information is stored in state.

**Answer: B ([LEAVE A REPLY](#))**

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 29**

True or False? `terraform init` cannot automatically download Community providers.

- A. False
- B. True

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 30**

Taint the resource "aws\_instance" "baz" resource that lives in module bar which lives in module foo.

- A. `terraform taint module.foo.module.bar.baz`
- B. `terraform taint module.foo.bar.aws_instance.baz`
- C. `terraform taint module.foo.module.bar.aws_instance.baz`
- D. `terraform taint foo.bar.aws_instance.baz`

**Answer: ([SHOW ANSWER](#))**

Check resource addressing

<https://www.terraform.io/docs/internals/resource-addressing.html>

### **NEW QUESTION: 31**

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 verbose logging for each provider in your Terraform configuration
- B. Set the environment variable `TF_VAR_log=TRACE`
- C. Set the environment variable `TF_LOG=TRACE`
- D. Set the environment variable `TF_LOG_PATH`

**Answer: C ([LEAVE A REPLY](#))**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

### NEW QUESTION: 32

A single terraform resource file that defines an aws\_instance resource can simply be renamed to vsphere\_virtual\_machine in order to switch cloud providers.

- A. True
- B. False

**Answer: B (LEAVE A REPLY)**

Every provider has its own required and allowed declarations none of which match between cloud providers.

### NEW QUESTION: 33

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

- A. True
- B. False

**Answer: A (LEAVE A REPLY)**

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

### NEW QUESTION: 34

Complete the following sentence:

The terraform state command can be used to \_\_\_\_\_

- A. modify state
- B. view state
- C. refresh state
- D. There is no such command

**Answer: (SHOW ANSWER)**

Explanation

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

### NEW QUESTION: 35

Command terraform refresh will update state file?

- A. False
- B. True

**Answer: (SHOW ANSWER)**

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. If the state is changed, this may cause changes to occur during the next plan or apply.

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

### NEW QUESTION: 36

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

**Answer: B,D** ([LEAVE A REPLY](#))

The `terraform import` command is used to import existing infrastructure. Import existing Google Cloud resources into Terraform with Terraformer.

### NEW QUESTION: 37

A user runs `terraform init` on their RHEL based server and per the output, two provider plugins are downloaded: \$ `terraform init` Initializing the backend...

Initializing provider plugins...

- Checking for available provider plugins...
  - Downloading plugin for provider "aws" (hashicorp/aws) 2.44.0...
  - Downloading plugin for provider "random" (hashicorp/random) 2.2.1...
- :

Terraform has been successfully initialized! Where are these plugins downloaded to?

- A. The `.terraform/plugins` directory in the directory `terraform init` was executed in.
- B. The `.terraform.plugins` directory in the directory `terraform init` was executed in.
- C. `/etc/terraform/plugins`
- D. The `.terraform.d` directory in the directory `terraform init` was executed in.

**Answer: A** ([LEAVE A REPLY](#))

### NEW QUESTION: 38

Terraform import command can import resources into modules as well directly into the root of your state.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

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 directly into the root of your state.

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The `terraform import` command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform. For example:

```
resource "aws_instance" "import_example" {  
  # ...instance configuration...  
}
```

Now `terraform import` can be run to attach an existing instance to this resource configuration:

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID `i-03efafa258104165f` (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name `aws_instance.import_example` in the Terraform state.

As a result of the above command, the resource is recorded in the state file. We can now run `terraform plan` to see how the configuration compares to the imported resource, and make any adjustments to the configuration to align with the current (or desired) state of the imported object.

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

## NEW QUESTION: 39

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`

**Answer: B ([LEAVE A REPLY](#))**

terraform init command is used to initialize a working directory containing Terraform configuration files. Reference: <https://www.terraform.io/docs/cli/commands/init.html>

#### **NEW QUESTION: 40**

Which of the following is not an action performed by terraform init?

- A. Load required provider plugins
- B. Initialize a configured backend
- C. Retrieve the source code for all referenced modules
- D. Create a sample main.tf file

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 41**

Using multi-cloud and provider-agnostic tools provides which of the following benefits?

- A. Operations teams only need to learn and manage a single tool to manage infrastructure, regardless of where the infrastructure is deployed.
- B. Increased risk due to all infrastructure relying on a single tool for management.
- C. Can be used across major cloud providers and VM hypervisors.
- D. Slower provisioning speed allows the operations team to catch mistakes before they are applied.

**Answer:** A,C ([LEAVE A REPLY](#))

Using a tool like Terraform can be advantageous for organizations deploying workloads across multiple public and private cloud environments. Operations teams only need to learn a single tool, single language, and can use the same tooling to enable a DevOps-like experience and workflows.

#### **NEW QUESTION: 42**

Which of the following challenges would Terraform be a candidate for solving? (Select THREE)

- A. Utilize a single tool for all of the infrastructure and configuration management needs.
- B. Enable self-service infrastructure to allocate resources on your proprietary private cloud.
- C. Have a single interoperable tool to manage the variety of services including GitHub repositories, MySQL database, and Kubernetes clusters.
- D. Reduce the number of workflows needed for managing infrastructure across each of the companies public and private clouds.

**Answer:** B,C,D ([LEAVE A REPLY](#))

#### **NEW QUESTION: 43**

When using remote state, state is only ever held in memory when used by Terraform.

- A. False
- B. True

**Answer:** B ([LEAVE A REPLY](#))

## **NEW QUESTION: 44**

Environment variables can be used to set variables. The environment variables must be in the format "\_\_\_\_\_"\_<variablename>. Select the correct prefix string from the following list.

- A. TF\_CLI\_ARGS
- B. TF\_VAR
- C. TF\_VAR\_
- D. TF\_VAR\_ENV

**Answer: C ([LEAVE A REPLY](#))**

Environment variables can be used to set variables. The environment variables must be in the format TF\_VAR\_name and this will be checked last for a value. For example:

```
export TF_VAR_region=us-west-1  
export TF_VAR_ami=ami-049d8641  
export TF_VAR_alist='[1,2,3]'  
export TF_VAR_amap='{ foo = "bar", baz = "qux" }'
```

<https://www.terraform.io/docs/commands/environment-variables.html>

## **NEW QUESTION: 45**

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

- A. terraform providers
- B. terraform state show
- C. terraform provider
- D. terraform plan

**Answer: A ([LEAVE A REPLY](#))**

Use the terraform providers command to view the specified version constraints for all providers used in the current configuration.

<https://www.terraform.io/docs/configuration/providers.html>

## **NEW QUESTION: 46**

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

**Answer: B ([LEAVE A REPLY](#))**

The variable declaration can also include a default argument.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

#### **NEW QUESTION: 47**

If you enable TF\_LOG = DEBUG, the log will be stored in syslog.log file in the current directory.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

<https://www.terraform.io/docs/internals/debugging.html>

#### **NEW QUESTION: 48**

When you initialize Terraform, where does it cache modules from the public Terraform Module Registry?

- A. In memory
- B. On disk in the /tmp directory
- C. On disk in the .terraform sub-directory
- D. They are not cached

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 49**

Which statement describes a goal of infrastructure as code?

- A. Write once, run anywhere
- B. An abstraction from vendor specific APIs
- C. The programmatic configuration of resources
- D. A pipeline process to test and deliver software

**Answer: C** ([LEAVE A REPLY](#))

#### **NEW QUESTION: 50**

Which option can not be used to keep secrets out of Terraform configuration files?

- A. A Terraform provider
- B. A -var flag
- C. Environment variables
- D. secure string

**Answer: B** ([LEAVE A REPLY](#))

## **NEW QUESTION: 51**

Which of the following clouds does not have a provider maintained HashiCorp?

- A.** IBM Cloud
- B.** DigitalOcean
- C.** OpenStack
- D.** AWS

**Answer:** A ([LEAVE A REPLY](#))

IBM Cloud does not have a provider maintained by HashiCorp, although IBM Cloud does maintain their own Terraform provider.

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

## **NEW QUESTION: 52**

Why might a user opt to include the following snippet in their configuration file?

- A.** Terraform 0.12 introduced substantial changes to the syntax used to write Terraform configuration
- B.** versions before Terraform 0.12 were not approved by HashiCorp to be used in production
- C.** The user wants to ensure that the application being deployed is a minimum version of 0.12
- D.** this ensures that all Terraform providers are above a certain version to match the application being deployed

**Answer:** ([SHOW ANSWER](#))

## **NEW QUESTION: 53**

A data block requests that Terraform read from a given data source and export the result under the given local name.

- A.** True
- B.** False

**Answer:** A ([LEAVE A REPLY](#))

## **NEW QUESTION: 54**

A "backend" in Terraform determines how state is loaded and how an operation such as apply is executed. Which of the following is not a supported backend type?

- A.** Terraform enterprise
- B.** Consul
- C.** Github
- D.** S3
- E.** Artifactory

**Answer:** ([SHOW ANSWER](#))

Github is not a supported backend type.

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

## **NEW QUESTION: 55**

Which of the following value will be accepted for var1?

```
variable "var1" {  
  type = string  
}
```

- A. None of the above
- B. Both A and B
- C. "5"
- D. 5

**Answer:** B ([LEAVE A REPLY](#))

Terraform automatically converts number and bool values to strings when needed.

## **NEW QUESTION: 56**

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.

**Answer:**

outFILENAME

## **NEW QUESTION: 57**

Terraform Enterprise currently supports running under which the following operating systems?

- A. Ubuntu
- B. Amazon Linux
- C. Debian
- D. CentOS
- E. Red Hat Enterprise Linux
- F. Oracle Linux

**Answer:** A,B,C,D,E,F ([LEAVE A REPLY](#))

Terraform Enterprise runs on Linux instances, and you must prepare a running Linux instance for Terraform Enterprise before running the installer. You will start and manage this instance like any other server.

Terraform Enterprise currently supports running under the following operating systems:

Standalone deployment:

Debian 7.7+

Ubuntu 14.04.5 / 16.04 / 18.04

Red Hat Enterprise Linux 7.4 - 7.8

CentOS 6.x / 7.4 - 7.8

Amazon Linux 2014.03 / 2014.09 / 2015.03 / 2015.09 / 2016.03 / 2016.09 / 2017.03 / 2017.09 / 2018.03 / 2.0 Oracle Linux 7.4 - 7.8

<https://www.terraform.io/docs/enterprise/before-installing/index.html>

### NEW QUESTION: 58

```
1. resource "aws_s3_bucket" "example" {  
2.   bucket = "my-test-s3-terraform-bucket"  
3. }  
4. resource "aws_iam_role" "test_role" {  
5.   name = "test_role"  
6. }
```

Due to the way that the application code is written, the s3 bucket must be created before the test role is created, otherwise there will be a problem. How can you ensure that?

- A. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- B. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- C. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- D. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

### Answer: A ([LEAVE A REPLY](#))

Implicit dependency works only if there is some reference of one resource to another.

Explicit dependency is the option here.

### NEW QUESTION: 59

You have declared a variable name my\_var in terraform configuration without a value associated with it.

```
variable my_var {}
```

After running terraform plan it will show an error as variable is not defined.

- A. True
- B. False

### Answer: B ([LEAVE A REPLY](#))

Input variables are usually defined by stating a name, type and a default value. However, the type and default values are not strictly necessary. Terraform can deduct the type of the variable from the default or input value.

Variables can be predetermined in a file or included in the command-line options. As such, the simplest variable is just a name while the type and value are selected based on the input.

```
variable "variable_name" {}  
terraform apply -var variable_name="value"
```

The input variables, like the one above, use a couple of different types: strings, lists, maps, and boolean. Here are some examples of how each type are defined and used.

String

Strings mark a single value per structure and are commonly used to simplify and make complicated values more user-friendly. Below is an example of a string variable definition.

```
variable "template" {
  type = string
  default = "01000000-0000-4000-8000-000030080200"
}
```

A string variable can then be used in resource plans. Surrounded by double quotes, string variables are a simple substitution such as the example underneath.

```
storage = var.template
```

## List

Another type of Terraform variables lists. They work much like a numbered catalogue of values. Each value can be called by their corresponding index in the list. Here is an example of a list variable definition.

```
variable "users" {
  type = list
  default = ["root", "user1", "user2"]
}
```

Lists can be used in the resource plans similarly to strings, but you'll also need to denote the index of the value you are looking for.

```
username = var.users[0]
```

## Map

Maps are a collection of string keys and string values. These can be useful for selecting values based on predefined parameters such as the server configuration by the monthly price.

```
variable "plans" {
  type = map
  default = {
    "5USD" = "1xCPU-1GB"
    "10USD" = "1xCPU-2GB"
    "20USD" = "2xCPU-4GB"
  }
}
```

You can access the right value by using the matching key. For example, the variable below would set the plan to "1xCPU-1GB".

```
plan = var.plans["5USD"]
```

The values matching to their keys can also be used to look up information in other maps. For example, underneath is a shortlist of plans and their corresponding storage sizes.

```
variable "storage_sizes" {
  type = map
  default = {
    "1xCPU-1GB" = "25"
  }
}
```

```
"1xCPU-2GB" = "50"
"2xCPU-4GB" = "80"
}
}
```

These can then be used to find the right storage size based on the monthly price as defined in the previous example.

```
size = lookup(var.storage_sizes, var.plans["5USD"])
```

Boolean

The last of the available variable type is boolean. They give the option to employ simple true or false values. For example, you might wish to have a variable that decides when to generate the root user password on a new deployment.

```
variable "set_password" {
  default = false
}
```

The above example boolean can be used similarly to a string variable by simply marking down the correct variable.

```
create_password = var.set_password
```

By default, the value is set to false in this example. However, you can overwrite the variable at deployment by assigning a different value in a command-line variable.

```
terraform apply -var set_password=true
```

## NEW QUESTION: 60

Which one is the right way to import a local module names consul?

- A. module "consul" { source = "consul"}
- B. module "consul" { source = "./consul"}
- C. module "consul" { source = "../consul"}
- D. module "consul" { source = "module/consul"}

**Answer: B,C ([LEAVE A REPLY](#))**

A local path must begin with either ./ or ../ to indicate that a local path is intended, to distinguish from a module registry address.

```
module "consul" {
  source = "./consul"
}
```

## NEW QUESTION: 61

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

1. max
2. min
3. join
4. replace

5. list
  6. length
  7. range
- A.** Backends  
**B.** Data sources  
**C.** Named values  
**D.** Functions

**Answer:** D ([LEAVE A REPLY](#))

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not support user-defined functions, and only the functions built into the language are available for use.

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 62

You want to use terraform import to start managing infrastructure that was not originally provisioned through infrastructure as code. Before you can import the resource's current state, what must you do in order to prepare to manage these resources using Terraform?

- A.** Run terraform refresh to ensure that the state file has the latest information for existing resources.  
**B.** Update the configuration file to include the new resources.  
**C.** Shut down or stop using the resources being imported so no changes are inadvertently missed.  
**D.** Modify the Terraform state file to add the new resources.

**Answer:** B ([LEAVE A REPLY](#))

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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. The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

### **NEW QUESTION: 63**

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

**Answer: C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 64**

What is the purpose of using the local-exec provisioner? (Select Two)

- A. To invoke a local executable.
- B. Executes a command on the resource to invoke an update to the Terraform state.
- C. To execute one or more commands on the machine running Terraform.

- D. Ensures that the resource is only executed in the local infrastructure where Terraform is deployed.

**Answer: A,C ([LEAVE A REPLY](#))**

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.

Note that even though the resource will be fully created when the provisioner is run, there is no guarantee that it will be in an operable state - for example system services such as sshd may not be started yet on compute resources.

Example usage

```
resource "aws_instance" "web" {  
# ...  
provisioner "local-exec" {  
command = "echo ${aws_instance.web.private_ip} >> private_ips.txt"  
}  
}
```

Note: Provisioners should only be used as a last resort. For most common situations there are better alternatives.

<https://www.terraform.io/docs/provisioners/local-exec.html>

**NEW QUESTION: 65**

What is a downside to using the Vault provider to read secrets from Vault?

- A. Secrets are persisted to the state file and plans.
- B. Terraform and Vault must be running on the same version.
- C. Terraform and Vault must be running on the same physical host.
- D. Terraform requires a unique auth method to work with Vault.

**Answer: A ([LEAVE A REPLY](#))**

The Vault provider allows Terraform to read from, write to, and configure Hashicorp Vault. Interacting with Vault from Terraform causes any secrets that you read and write to be persisted in both Terraform's state file and in any generated plan files. For any Terraform module that reads or writes Vault secrets, these files should be treated as sensitive and protected accordingly.

**NEW QUESTION: 66**

You wanted to destroy some of the dependent resources from real infrastructure. You choose to delete those resources from your configuration file and run `terraform plan` and then `apply`. Which of the following way your resources would be destroyed?

- A. Terraform can still determine the correct order for destruction from the state even when you delete one or more items from the configuration.
- B. Those would be destroyed in the order in which they were written in the configuration file previously before you have deleted them from configuration file.

C. The resource will be destructed in random order as you have already deleted them from configuration.

D. You can not destroy resources by deleting them from configuration file and running plan and apply.

**Answer: A ([LEAVE A REPLY](#))**

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state. Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

### **NEW QUESTION: 67**

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

A. True

B. False

**Answer: A ([LEAVE A REPLY](#))**

### **NEW QUESTION: 68**

What happens when a terraform apply command is executed?

A. Creates the execution plan for the deployment of resources.

B. Applies the changes required in the target infrastructure in order to reach the desired configuration.

C. The backend is initialized and the working directory is prepped.

D. Reconciles the state Terraform knows about with the real-world infrastructure.

**Answer: B ([LEAVE A REPLY](#))**

The terraform apply command is used to apply the changes required to reach the desired state of the configuration, or the pre-determined set of actions generated by a terraform plan execution plan.

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

### **NEW QUESTION: 69**

Provisioners should only be used as a last resort.

A. False

B. True

**Answer: B ([LEAVE A REPLY](#))**

Provisioners are a Last Resort

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.

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

### **NEW QUESTION: 70**

Which one of the following command will rewrite Terraform configuration files to a canonical format and style.

- A. `terraform graph -h`
- B. `terraform init`
- C. `terraform graph`
- D. `terraform fmt`

**Answer: D ([LEAVE A REPLY](#))**

The `terraform fmt` command is used to rewrite Terraform configuration files to a canonical format and style. This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability.

### **NEW QUESTION: 71**

A Terraform provider is not responsible for:

- A. Understanding API interactions with some service
- B. Managing actions to take based on resource differences
- C. Exposing resources and data sources based on an API
- D. Provisioning infrastructure in multiple clouds

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 72**

Complete the following sentence:

For local state, the workspaces are stored directly in a \_\_\_\_\_.

- A. a file called `terraform.tfstate.backup`

- B. directory called `terraform.workspaces.tfstate`
- C. a file called `terraform.tfstate`
- D. directory called `terraform.tfstate.d`

**Answer:** ([SHOW ANSWER](#))

For local state, Terraform stores the workspace states in a directory called `terraform.tfstate.d`.

<https://www.terraform.io/docs/state/workspaces.html#workspace-internals>

### NEW QUESTION: 73

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]`

**Answer:** B ([LEAVE A REPLY](#))

### NEW QUESTION: 74

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address `aws_instance.web` refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.
- C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.
- D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer:** ([SHOW ANSWER](#))

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:  
`module.A.module.B.module.C...`

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

`aws_instance.web[3]`

Refers to only the last instance in the config, and an address like this:

`aws_instance.web`

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

## NEW QUESTION: 75

Jim has created several AWS resources from a single terraform configuration file.

Someone from his team has manually modified one of the EC2 instance.

Now to discard the manual change, Jim wants to destroy and recreate the EC2 instance.

What is the best way to do it?

- A. `terraform recreate`
- B. `terraform taint`
- C. `terraform destroy`
- D. `terraform refresh`

**Answer: B ([LEAVE A REPLY](#))**

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially new IP address of a tainted server. The `plan` command will show this if this is the case.

This example will taint a single resource:

```
$ terraform taint aws_security_group.allow_all
```

The resource aws\_security\_group.allow\_all in the module root has been marked as tainted.

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

### NEW QUESTION: 76

You have created two workspaces PROD and DEV. You have switched to DEV and provisioned DEV infrastructure from this workspace. Where is your state file stored?

- A. terraform.d
- B. terraform.tfstate
- C. terraform.tfstate.DEV
- D. terraform.tfstate.d

**Answer: D ([LEAVE A REPLY](#))**

Terraform stores the workspace states in a directory called terraform.tfstate.d. This directory should be treated similarly to default workspace state file terraform.tfstate

```
└── main.tf
└── provider.tf
└── terraform.tfstate.d
    ├── DEV
    │   └── terraform.tfstate # DEV workspace state file
    └── PROD
        └── terraform.tfstate # PROD workspace state file
└── terraform.tfvars # Default workspace state file
└── variables.tf
```

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

[\(324 Q&As Dumps, 30%OFF](https://www.fast2test.com/TA-002-P-premium-file.html)

**Special Discount: freecram)**

### NEW QUESTION: 77

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. [ for o in var.list : o.id ]
- C. [ var.list[\*].id ]
- D. var.list[\*].id

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 78**

Which of the following type of variable allows multiple values of several distinct types to be grouped together as a single value?

- A. Map
- B. Object
- C. Tuple
- D. List

**Answer:** ([SHOW ANSWER](#))

Structural type of variable allows multiple values of several distinct types to be grouped together as a single value. They require a schema as an argument, to specify which types are allowed for which elements.

<https://www.terraform.io/docs/configuration/types.html>

### **NEW QUESTION: 79**

Only the user that generated a plan may apply it.

- A. True
- B. False

**Answer:** A ([LEAVE A REPLY](#))

The optional -out argument can be used to save the generated plan to a file for later execution with terraform apply, which can be useful when running Terraform in automation.

### **NEW QUESTION: 80**

Terraform and Terraform providers must use the same major version number in a single configuration.

- A. False
- B. True

**Answer:** A ([LEAVE A REPLY](#))

### **NEW QUESTION: 81**

Mary has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code:

1. output "db\_password"
2. {
3. value = local.db\_password
4. }

Mary wants to hide the output value in the CLI after terraform apply? What is the best way?

- A. Use secure parameter
- B. Use sensitive parameter
- C. Encrypt the value using encrypt() function

D. Use cryptographic hash

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 82

Which of the following value will be accepted for my\_var?

1. variable "my\_var"

2. {

3. type = string

4. }

A. 15

B. "15"

C. Both A and B

D. None of the above

**Answer:** ([SHOW ANSWER](#))

The Terraform language will automatically convert number and bool values to string values when needed, and vice-versa as long as the string contains a valid representation of a number or boolean value.

Example

\* true converts to "true", and vice-versa

\* false converts to "false", and vice-versa

\* 15 converts to "15", and vice-versa

Where possible, Terraform automatically converts values from one type to another in order to produce the expected type. If this isn't possible, Terraform will produce a type mismatch error and you must update the configuration with a more suitable expression.

<https://www.terraform.io/docs/configuration/expressions.html#type-conversion>

### NEW QUESTION: 83

The terraform state command can be used to \_\_\_\_\_

A. Update current state

B. Refresh existing state file

C. Print the current state file in console

D. It is not a valid command

**Answer:** ([SHOW ANSWER](#))

The terraform state command is used for advanced state management. Rather than modify the state directly, the terraform state commands can be used in many cases instead.

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

### NEW QUESTION: 84

Dawn has created the below child module. Without changing the module, can she override the instance\_type from t2.micro to t2.large from her code while calling this module?

1. resource "aws\_instance" "myec2"

```
2. {  
3.   ami = "ami-082b5a644766e0e6f"  
4.   instance_type = "t2.micro"  
5. }
```

- A. YES
- B. No

**Answer: B ([LEAVE A REPLY](#))**

As the instance\_type is hard-coded in source module, you will not be able to change its value from destination module. Instead of hard-coding you should use variable with default values.

**NEW QUESTION: 85**

When should you use the force-unlock command?

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

**Answer: C ([LEAVE A REPLY](#))**

Manually unlock the state for the defined configuration.

**NEW QUESTION: 86**

Which of the following is not a valid Terraform collection type?

- A. list
- B. map
- C. set
- D. tree

**Answer: D ([LEAVE A REPLY](#))**

**NEW QUESTION: 87**

Terraform provisioners can be added to any resource block.

- A. False
- B. True

**Answer: B ([LEAVE A REPLY](#))**

**NEW QUESTION: 88**

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 state
- B. terraform import
- C. terraform init

D. terraform workspace

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 89**

Terraform provisioners that require authentication can use the \_\_\_\_\_ block.

- A. ssh
- B. credentials
- C. connection
- D. secrets

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 90**

A terraform apply can not \_\_\_\_\_ infrastructure.

- A. import
- B. change
- C. destroy
- D. provision

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 91**

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 state list to find the name of the resource, then terraform state show to find the attributes including public IP address
- B. Run terraform destroy then terraform apply and look for the IP address in stdout
- C. 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
- D. Run terraform output ip\_address to view the result

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

#### **NEW QUESTION: 92**

Ric wants to enable detail logging and he wants highest verbosity of logs. Which of the following environment variable settings is correct option for him to select.

- A. Set TF\_LOG = DEBUG
- B. Set VAR\_TF = TRACE
- C. Set TF\_LOG = TRACE
- D. Set VAR\_TF\_LOG = TRACE

**Answer: C ([LEAVE A REPLY](#))**

<https://www.terraform.io/docs/internals/debugging.html>

### **NEW QUESTION: 93**

You write a new Terraform configuration and immediately run `terraform apply` in the CLI using the local backend.

Why will the apply fail?

- A. The Terraform CLI needs you to log into Terraform cloud first
- B. Terraform needs to install the necessary plugins first
- C. Terraform needs you to format your code according to best practices first
- D. Terraform requires you to manually run `terraform plan` first

**Answer: A ([LEAVE A REPLY](#))**

### **NEW QUESTION: 94**

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

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

### **NEW QUESTION: 95**

Setting the `TF_LOG` environment variable to `DEBUG` causes debug messages to be logged into syslog.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

### **NEW QUESTION: 96**

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 plan` 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

**Answer: C ([LEAVE A REPLY](#))**

changes, and terraform apply to create new infrastructure.

Reference:

+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply  
+to+create+new  
+infrastructure.&oq=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run  
+terraform+plan+to  
+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new  
+infrastructure.&aqs=chrome..69i57.556j0j7&sourceid=chrome&ie=UTF-8

### **NEW QUESTION: 97**

Module variable assignments are inherited from the parent module and do not need to be explicitly set.

A. False

B. True

**Answer: A ([LEAVE A REPLY](#))**

### **NEW QUESTION: 98**

HashiCorp offers multiple versions of Terraform, including Terraform open-source, Terraform Cloud, and Terraform Enterprise. Which of the following Terraform features are only available in the Enterprise edition? (select four)

A. SAML/SSO

B. Sentinel

C. Audit Logs

D. Clustering

E. Private Module Registry

F. Private Network Connectivity

**Answer: ([SHOW ANSWER](#))**

Explanation

While there are a ton of features that are available to open source users, many features that are part of the Enterprise offering are geared towards larger teams and enterprise functionality. To see what specific features are part of Terraform Cloud and Terraform Enterprise, check out this link. <https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 99**

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

A. Outputs

B. Required input variables

- C. Optional inputs variables and default values
- D. All of the above
- E. None of the above

**Answer: E ([LEAVE A REPLY](#))**

### **NEW QUESTION: 100**

Which Terraform command will force a marked resource to be destroyed and recreated on the next apply?

- A. terraform fmt
- B. terraform destroy
- C. terraform taint
- D. terraform refresh

**Answer: C ([LEAVE A REPLY](#))**

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially new IP address of a tainted server. The plan command will show this if this is the case.

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

### **NEW QUESTION: 101**

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

D. Add version = "1.0.0" attribute to module block

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 102**

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

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

**Answer: A ([LEAVE A REPLY](#))**

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>

### **NEW QUESTION: 103**

You have written a terraform IaC script which was working till yesterday , but is giving some vague error from today , which you are unable to understand . You want more detailed logs that could potentially help you troubleshoot the issue , and understand the root cause. What can you do to enable this setting? Please note , you are using terraform OSS.

- A. Terraform OSS can push all its logs to a syslog endpoint. As such, you have to set up the syslog sink, and enable TF\_LOG\_PATH env variable to the syslog endpoint and all logs will automatically start streaming.
- B. Detailed logs are not available in terraform OSS, except the crash message. You need to upgrade to terraform enterprise for this point.
- C. Enable the TF\_LOG\_PATH to the log sink file location, and logging output will automatically be stored there.
- D. Enable TF\_LOG to the log level DEBUG, and then set TF\_LOG\_PATH to the log sink file location. Terraform debug logs will be dumped to the sink path, even in terraform OSS.

**Answer: D ([LEAVE A REPLY](#))**

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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

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.

### **NEW QUESTION: 104**

A single terraform resource file that defines an aws\_instance resource can simple be renamed to azurerm\_virtual\_machine in order to switch cloud providers

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

Providers usually require some configuration of their own to specify endpoint URLs, regions, authentication settings.

Providers Initialization can be done by either explicitly via a provider block or by adding a resource from that provider

<https://www.terraform.io/docs/configuration/providers.html>

### **NEW QUESTION: 105**

If a module declares a variable with a default, that variable must also be defined within the module.

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 106**

You want to get involved in the development of Terraform. As this is an open source project, you would like to contribute a fix for an open issue of Terraform. What programming language will need to use to write the fix?

- A. It depends on which command issue related to.
- B. Python
- C. Go
- D. Java

**Answer: C ([LEAVE A REPLY](#))**

Basic programming knowledge. Terraform and Terraform Plugins are written in the Go programming language, but even if you've never written a line of Go before, you're still welcome to take a dive into the code and submit patches. The community is happy to assist with code reviews and offer guidance specific to Go.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

[\(324 Q&As Dumps, 30%OFF](https://www.fast2test.com/TA-002-P-premium-file.html)

**Special Discount: freecram**)

## **NEW QUESTION: 107**

terraform validate validates the syntax of Terraform files.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

The terraform validate command validates the syntax and arguments of the Terraform configuration files. Reference: <https://www.terraform.io/docs/cli/code/index.html>

## **NEW QUESTION: 108**

Which task does terraform init not perform?

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

**Answer: ([SHOW ANSWER](#))**

## **NEW QUESTION: 109**

Which of the following connection types are supported by the remote-exec provisioner?

(select two)

- A. WinRM
- B. UDP
- C. SMB
- D. RDP
- E. ssh

**Answer: A,E ([LEAVE A REPLY](#))**

The remote-exec provisioner invokes a script on a remote resource after it is created. The remote-exec provisioner supports both ssh and winrm type connections.

remote-exec connection types -

- \* ssh on Linux
- \* winrm on Windows

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

## **NEW QUESTION: 110**

You have created a custom variable definition file testing.tfvars. How will you use it for provisioning infrastructure?

- A. terraform apply -var-state-file ="testing.tfvars"
- B. terraform plan -var-file="testing.tfvar"
- C. terraform apply -var-file="testing.tfvars"
- D. terraform apply var-file="testing.tfvars"

**Answer: C ([LEAVE A REPLY](#))**

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

### NEW QUESTION: 111

Which of the following Terraform files should be ignored by Git when committing code to a repo? (select Three)

- A. Files named exactly `terraform.tfvars` or `terraform.tfvars.json`.
- B. Any files with names ending in `.auto.tfvars` or `.auto.tfvars.json`.
- C. `input.tf`
- D. `terraform.tfstate`
- E. `output.tf`

**Answer:** ([SHOW ANSWER](#))

The `.gitignore` file should be configured to ignore Terraform files that either contain sensitive data or are not required to save.

Terraform state (`terraform.tfstate`) can contain sensitive data, depending on the resources in use and your definition of "sensitive." The state contains resource IDs and all resource attributes. For resources such as databases, this may contain initial passwords.

When using local state, state is stored in plain-text JSON files.

The `terraform.tfvars` file may contain sensitive data, such as passwords or IP addresses of an environment that you may not want to share with others.

### NEW QUESTION: 112

What kind of resource dependency is stored in `terraform.tfstate` file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

**Answer:** ([SHOW ANSWER](#))

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### NEW QUESTION: 113

Which of the below options is a valid interpolation syntax for retrieving a data source?

- A.  `${google_storage_bucket.backend}`
- B.  `${azurerm_resource_group.test.data}`
- C.  `${aws_instance.web.id.data}`
- D.  `${data.google_dns_keys.foo_dns_keys.key_signing_keys[0].ds_record}`

**Answer:** D ([LEAVE A REPLY](#))

Data source attributes are interpolated with the general syntax  
data.TYPE.NAME.ATTRIBUTE. The interpolation for a resource is the same but without  
the data. prefix (TYPE.NAME.ATTRIBUTE).

<https://www.terraform.io/docs/configuration-0-11/interpolation.html#attributes-of-a-data-source>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

[\(324 Q&As Dumps, 30%OFF](https://www.fast2test.com/TA-002-P-premium-file.html)

**Special Discount: freecram)**

**NO.1** What features stops multiple admins from changing the Terraform state at the same time?

- A.** Provider constraints
- B.** State locking
- C.** Backend types
- D.** Version control

**Answer:** B

Explanation

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>

**NO.2** Which of the following is true about terraform apply? (Choose two.)

- A.** You cannot target specific resources for the operation
- B.** By default, it does not refresh your state file to reflect current infrastructure configuration
- C.** Depending on provider specification, Terraform may need to destroy and recreate your infrastructure resources
- D.** You must pass the output of a terraform plan command to it
- E.** It only operates on infrastructure defined in the current working directory or workspace

**Answer:** C,E

Explanation

<https://www.terraform.io/cli/run>

**NO.3** 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\_10G\_LEVEL
- B.** TF\_LOG
- C.** TF\_10G\_PATM
- D.** TF.LOG.FUE

**Answer:** B

Explanation

<https://www.terraform.io/internals/debugging>

**NO.4** Changing the Terraform backend from the default "local" backend to a different one after doing your first terraform apply is:

- A.** Optional
- B.** Impossible
- C.** Mandatory

**D.** Discouraged

**Answer:** A

**NO.5** Terraform Enterprise currently supports running under which the following operating systems?

- A.** Ubuntu
- B.** Red Hat Enterprise Linux
- C.** Debian
- D.** Amazon Linux
- E.** Oracle Linux
- F.** CentOS

**Answer:** A,B,C,D,E,F

Explanation

Terraform Enterprise runs on Linux instances, and you must prepare a running Linux instance for Terraform

Enterprise before running the installer. You will start and manage this instance like any other server.

Terraform Enterprise currently supports running under the following operating systems:

Standalone deployment:

Debian 7.7+

Ubuntu 14.04.5 / 16.04 / 18.04

Red Hat Enterprise Linux 7.4 - 7.8

CentOS 6.x / 7.4 - 7.8

Amazon Linux 2014.03 / 2014.09 / 2015.03 / 2015.09 / 2016.03 / 2016.09 / 2017.03 / 2017.09 / 2018.03 / 2.0

Oracle Linux 7.4 - 7.8

<https://www.terraform.io/docs/enterprise/before-installing/index.html>

**NO.6** What does terraform plan do ?

- A.** Create an execution plan by evaluating the difference between configuration file and actual infrastructure.
- B.** Create an execution plan by evaluating the difference between configuration file and state file.
- C.** Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.
- D.** Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.

**Answer:** B

**NO.7** Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A.** Use of any resource belonging to a particular provider in a resource or data block in configuration.
- B.** Explicit use of a provider block in configuration, optionally including a version constraint.
- C.** Existence of any provider plugins found locally in the working directory.

**D.** Existence of any resource instance belonging to a particular provider in the current state.

**Answer:** A,B,D

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider dependency. The plugin can exist without any reference to it in the terraform configuration.

<https://www.terraform.io/docs/providers.html>

**NO.8** If a module uses a local variable, you can expose that value with a terraform output.

**A.** True

**B.** False

**Answer:** A

Explanation

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>

**NO.9** How would you reference the attribute "name\*" of this fictitious resource in HCL?

```
resource "kubernetes_namespace" "example" {
  name = "test"
}
```

**A.** data kubernetes\_namespace.name

**B.** kubernetes\_namespace.example.name

**C.** None of the above

**D.** kubernetes\_namespace.test.name

**E.** resource.kubrnetes\_namespace>example.name

**Answer:** B

Explanation

<https://www.terraform.io/language/expressions/references#references-to-resource-attributes>

**NO.10 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.

**Answer:**

-out=FILENAME

"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>

**NO.11** Mary has created a database instance in AWS and for ease of use is outputting the value of the database

password with the following code:

1. output "db\_password"
2. {
3. value = local.db\_password
4. }

Mary wants to hide the output value in the CLI after terraform apply? What is the best way?

- A.** Use secure parameter
- B.** Use sensitive parameter
- C.** Use cryptographic hash
- D.** Encrypt the value using encrypt() function

**Answer:** B

**NO.12** 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.id
- B.** aws\_ami.ubuntu
- C.** data.aws\_ami.ubuntu.id
- D.** data.aws\_ami.ubuntu

**Answer:** C

Explanation

```
resource "aws_instance" "web" {  
  ami= data.aws_ami.ubuntu.id  
Reference: https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance
```

**NO.13** You want to know from which paths Terraform is loading providers referenced in your Terraform configuration (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\_PATH
- B.** Set the environment variable TF\_LOG=TRACE
- C.** Set verbose logging for each provider in your Terraform configuration
- D.** Set the environment variable TF\_VAR\_log=TRACE

**Answer:** B

Explanation

Although this will only output to stderr and if you need to review log file you will need to include TF\_LOG\_PATH=pathToFile <https://www.terraform.io/internals/debugging>

**NO.14** Select the answer below that completes the following statement: Terraform Cloud can be managed from the

CLI but requires \_\_\_\_\_?

- A.** authentication using MFA
- B.** a TOTP token
- C.** a username and password
- D.** an API token

**Answer:** D

Explanation

API and CLI access are managed with API tokens, which can be generated in the Terraform Cloud UI.

Each

user can generate any number of personal API tokens, which allow access with their own identity and permissions. Organizations and teams can also generate tokens for automating tasks that aren't tied to an individual user.

**NO.15** When should Terraform configuration files be written when running `terraform import` on existing infrastructure?

- A.** Terraform will generate the corresponding configuration files for you
- B.** Terraform configuration should be written before `terraform import` is executed
- C.** You should write Terraform configuration files after the next `terraform import` is executed
- D.** Infrastructure can be imported without corresponding Terraform code

**Answer:** B

Explanation

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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>

**NO.16** Which of the following is the right substitute for static values that can make Terraform configuration file more dynamic and reusable?

- A.** Input parameters
- B.** Functions
- C.** Modules
- D.** Output value

**Answer:** A

Explanation

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module's own source code, and allowing modules to be shared between different configurations.

**NO.17** resource "aws\_s3\_bucket" "example" { bucket = "my-test-s3-terraform-bucket" ...} resource "aws\_iam\_role" "test\_role" { name = "test\_role" ...}

Due to the way that the application code is written , the s3 bucket must be created before the test role is created , otherwise there will be a problem. How can you ensure that?

- A.** This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- B.** This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- C.** Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.
- D.** Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.

**Answer:** D

Explanation

Use the depends\_on meta-argument to handle hidden resource dependencies that Terraform can't automatically infer.

Explicitly specifying a dependency is only necessary when a resource relies on some other resource's behavior but doesn't access any of that resource's data in its arguments.

**NO.18** You have created 2 workspaces PROD and RQA. You have switched to RQA and provisioned RQA

infrastructure from this workspace. Where is your state file stored?

- A.** terraform.tfstate.d
- B.** terraform.d
- C.** terraform.tfstate
- D.** terraform.tfstate.RQA

**Answer:** A

**NO.19** What is the standard workflow that a developer follows while working with terraform open source version?

- A.** Run terraform destroy first since you need to start from scratch every time , before running terraform apply.
- B.** Run terraform refresh to update the terraform state , then write the terraform code , and finally run terraform apply.
- C.** Write terraform code , and run terraform push , to update the terraform state to the remote repo , which in turn will take care of the next steps.
- D.** Write the terraform code on the developer machine , run terraform plan to check the changes , and run terraform apply to provision the infra.

**Answer:** D

Explanation

You do not need to run terraform refresh as terraform plan implicitly will run terraform refresh.

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

**NO.20** Which statement describes a goal of infrastructure as code?

- A.** An abstraction from vendor specific APIs
- B.** Write once, run anywhere
- C.** The programmatic configuration of resources
- D.** A pipeline process to test and deliver software

**Answer:** C

Explanation

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.

**NO.21** 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

**Answer:** D

**NO.22** Which of the following is not a benefit of adopting infrastructure as code?

- A. Interpolation
- B. Versioning
- C. Reusability of code
- D. Automation

**Answer:** A

**NO.23** When does terraform apply reflect changes in the cloud environment?

- A. Immediately
- B. After updating the state file
- C. Based on the value provided to the -refresh command line argument
- D. None of the above
- E. However long it takes the resource provider to fulfill the request

**Answer:** E

**NO.24** 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

**Answer:** B

**NO.25** When TF\_LOG\_PATH is set, TF\_LOG must be set in order for any logging to be enabled.

- A. True
- B. False

**Answer:** A

Explanation

TF\_LOG\_PATH specifies where the log should persist its output to. Note that even when TF\_LOG\_PATH is

set, TF\_LOG must be set in order for any logging to be enabled.

For example, to always write the log to the directory you're currently running terraform from:

```
export TF_LOG_PATH=./terraform.log  
export TF_LOG=TRACE
```

**NO.26** Your team uses terraform OSS . You have created a number of reusable modules for important , independent network components that you want to share with your team to enhance consistency . What is the correct option/way to do that?

- A.** Upload your modules with proper versioning in the terraform public module registry . Terraform OSS is directly integrated with the public module registry , and can reference the modules from the code in the main tf file.
- B.** Terraform modules cannot be shared in OSS version . Each developer needs to maintain their own modules and leverage them in the main tf file.
- C.** Terraform module sharing is only available in Enterprise version via terraform private module registry , so no way to enable it in OSS version.
- D.** Store your modules in a NAS/ shared file server , and ask your team members to directly reference the code from there. This is the only viable option in terraform OSS ,which is better than individually maintaining module versions for every developer.

**Answer:** A

Explanation

Software development encourages code reuse through reusable artifacts, such as libraries, packages and modules. Most programming languages enable developers to package and publish these reusable components and make them available on a registry or feed. For example, Python has Python Package Index and PowerShell has PowerShell Gallery. For Terraform users, the Terraform Registry enables the distribution of Terraform modules, which are reusable configurations. The Terraform Registry acts as a centralized repository for module sharing, making modules easier to discover and reuse.

The Registry is available in two variants:

- \* Public Registry houses official Terraform providers -- which are services that interact with an API to expose and manage a specific resource -- and community-contributed modules.
- \* Private Registry is available as part of the Terraform Cloud, and can host modules internally within an organization.

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

**NO.27** When multiple engineers start deploying infrastructure using the same state file, what is a

---

feature of remote state storage that is critical to ensure the state doesn't become corrupt?

- A. Object Storage
- B. Encryption
- C. State Locking
- D. WorkSpaces

**Answer:** C

Explanation

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.

State locking happens automatically on all operations that could write state. You won't see any message that it

is happening. If state locking fails, Terraform will not continue. You can disable state locking for most commands with the `-lock` flag but it is not recommended.

If acquiring the lock is taking longer than expected, Terraform will output a status message. If Terraform

doesn't output a message, state locking is still occurring if your backend supports it.

Not all backends support locking. Please view the list of backend types for details on whether a backend

supports locking or not.

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

**NO.28** When configuring a remote backend in Terraform, it might be a good idea to purposely omit some of the

required arguments to ensure secrets and other important data aren't inadvertently shared with others. What are

the ways the remaining configuration can be added to Terraform so it can initialize and communicate with the

backend? (select three)

- A. command-line key/value pairs
- B. interactively on the command line
- C. directly querying HashiCorp Vault for the secrets
- D. use the `-backend-config=PATH` to specify a separate config file

**Answer:** A,B,D

Explanation

You do not need to specify every required argument in the backend configuration. Omitting certain arguments

may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or

all of the arguments are omitted, we call this a partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process. There are several ways to supply the remaining arguments:

<https://www.terraform.io/docs/backends/init.html#backend-initialization>

**NO.29** True or False? Each Terraform workspace uses its own state file to manage the infrastructure associated with that particular workspace.

- A.** False
- B.** True

**Answer:** B

Explanation

The persistent data stored in the backend belongs to a workspace. Initially, the backend has only one workspace, called "default", and thus there is only one Terraform state associated with that configuration.

**NO.30** In the example below, the depends\_on argument creates what type of dependency?

- A.** internal dependency
- B.** implicit dependency
- C.** explicit dependency
- D.** non-dependency resource

**Answer:** C

# HashiCorp.TA-002-P.v2022-12-08.q137

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	137
<b>Version:</b>	v2022-12-08
<b># of views:</b>	105
<b># of Questions views:</b>	1370

<https://www.freecram.net/torrent/HashiCorp.TA-002-P.v2022-12-08.q137.html>

## NEW QUESTION: 1

Which of the following is not true of Terraform providers?

- A. Providers can be written by individuals
- B. Major cloud vendors and non-cloud vendors can write, maintain, or collaborate on Terraform providers
- C. Some providers are maintained by HashiCorp
- D. None of the above
- E. Providers can be maintained by a community of users

Answer: ([SHOW ANSWER](#))

## NEW QUESTION: 2

How does Terraform handle working with so many providers?

- A. Terraform ships with all of the plugins embedded in the Terraform binary.
- B. Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in the configuration's working directory.
- C. Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in a shared, system-wide plugins directory.
- D. Terraform allows you to select the providers you want to support during the Terraform installation process.

Answer: ([SHOW ANSWER](#))

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

## NEW QUESTION: 3

When using remote state, state is only ever held in memory when used by Terraform.

- A. False

B. True

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 4**

Choose the answer that correctly completes the sentence: \_\_\_\_\_ backends support state locking.

A. Only local

B. No

C. Some

D. All

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 5**

You have declared an input variable called environment in your parent module. What must you do to pass the value to a child module in the configuration?

A. Nothing, child modules inherit variables of parent module

B. Declare the variable in a terraform.tfvars file

C. Add node\_count = var.node\_count

D. Declare a node\_count input variable for child module

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 6**

The current implementation of Terraform import can only import resources into the state. It does not generate configuration.

A. False

B. True

**Answer:** ([SHOW ANSWER](#))

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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/docs/import/index.html#currently-state-only>

#### **NEW QUESTION: 7**

Terraform Enterprise currently supports running under which the following operating systems?

A. Ubuntu

B. Amazon Linux

C. Debian

D. CentOS

E. Red Hat Enterprise Linux

F. Oracle Linux

**Answer: ([SHOW ANSWER](#))**

Terraform Enterprise runs on Linux instances, and you must prepare a running Linux instance for Terraform Enterprise before running the installer. You will start and manage this instance like any other server.

Terraform Enterprise currently supports running under the following operating systems:

Standalone deployment:

Debian 7.7+

Ubuntu 14.04.5 / 16.04 / 18.04

Red Hat Enterprise Linux 7.4 - 7.8

CentOS 6.x / 7.4 - 7.8

Amazon Linux 2014.03 / 2014.09 / 2015.03 / 2015.09 / 2016.03 / 2016.09 / 2017.03 / 2017.09 /

2018.03 / 2.0 Oracle Linux 7.4 - 7.8

<https://www.terraform.io/docs/enterprise/before-installing/index.html>

## **NEW QUESTION: 8**

Which one of the following will run echo 0 and echo 1 on a newly created host?

- A. provisioner "local-exec" { command = "echo 0"command = "echo 1"}
- B. provisioner "remote-exec" {inline = [echo 0,echo 1]}
- C. provisioner "remote-exec" {command = "\${echo 0}"command = "\${echo 1}"}
- D. provisioner "remote-exec" {inline = ["echo 0","echo 1"]}

**Answer: ([SHOW ANSWER](#))**

remote-exec Provisioner

Example usage

```
resource "aws_instance" "web" {  
# ...  
provisioner "remote-exec" {  
inline = [  
"puppet apply",  
"consul join ${aws_instance.web.private_ip}",  
]  
}  
}
```

## **NEW QUESTION: 9**

John wants to use two different regions to deploy two different EC2 instances. He has specified two provider blocks in his providers.tf file.

```
provider "aws" { region = "us-east-1" }  
provider "aws" { region = "us-west-2" }
```

When he run terraform plan he encountered an error. How to fix this?

- A. Use alias for region = "us-west-2"
- B. It can not be fixed
- C. Use another provider version
- D. Use default keyword with region = "us-east-1"

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 10

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

**Answer:** B ([LEAVE A REPLY](#))

### NEW QUESTION: 11

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

- A. False

- B. True

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 12

While using generic git repository as a module source, which of the below options allows terraform to select a specific version or tag instead of selecting the HEAD.

- A. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?ref=v1.2.0"}
```

- B. Append version argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?version=v1.2.0"}
```

- C. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git#ref=v1.2.0"}
```

- D. By default, Terraform will clone and use the default branch (referenced by HEAD) in the selected repository and you can not override this.

**Answer:** ([SHOW ANSWER](#))

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:

```
module "vpc" {  
  source = "git::https://example.com/vpc.git?ref=v1.2.0"  
}
```

The value of the ref argument can be any reference that would be accepted by the git checkout command, including branch and tag names.

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

### **NEW QUESTION: 13**

What does the command terraform fmt do?

- A. Rewrite Terraform configuration files to a canonical format and style.
- B. Deletes the existing configuration file.
- C. Updates the font of the configuration file to the official font supported by HashiCorp.
- D. Formats the state file in order to ensure the latest state of resources can be obtained.

**Answer: ([SHOW ANSWER](#))**

The terraform fmt command is used to rewrite Terraform configuration files to a canonical format and style. This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability.

Other Terraform commands that generate Terraform configuration will produce configuration files that conform to the style imposed by terraform fmt, so using this style in your own files will ensure consistency.

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

### **NEW QUESTION: 14**

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

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

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 15**

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

- A. replace
- B. format
- C. join
- D. tostring

**Answer: ([SHOW ANSWER](#))**

<https://www.terraform.io/docs/configuration/functions/tostring.html>

### NEW QUESTION: 16

A user runs terraform init on their RHEL based server and per the output, two provider plugins are downloaded: \$ terraform init Initializing the backend...

Initializing provider plugins...

- Checking for available provider plugins...
- Downloading plugin for provider "aws" (hashicorp/aws) 2.44.0...
- Downloading plugin for provider "random" (hashicorp/random) 2.2.1...

:

Terraform has been successfully initialized! Where are these plugins downloaded to?

- A. The .terraform.d directory in the directory terraform init was executed in.
- B. The .terraform.plugins directory in the directory terraform init was executed in.
- C. The .terraform/plugins directory in the directory terraform init was executed in.
- D. /etc/terraform/plugins

Answer: ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 17

Which of the following type of variable allows multiple values of several distinct types to be grouped together as a single value?

- A. Map
- B. Object
- C. Tuple
- D. List

Answer: ([SHOW ANSWER](#))

Structural type of variable allows multiple values of several distinct types to be grouped together as a single value. They require a schema as an argument, to specify which types are allowed for which elements.

<https://www.terraform.io/docs/configuration/types.html>

### NEW QUESTION: 18

John is writing a module and within the module, there are multiple places where he has to use the same conditional expression but he wants to avoid repeating the same values or expressions multiple times in a configuration,. What is a better approach to dealing with this?

- A. Local Values
- B. Expressions
- C. Functions
- D. Variables

**Answer:** ([SHOW ANSWER](#))

A local value assigns a name to an expression, allowing it to be used multiple times within a module without repeating it.

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

### **NEW QUESTION: 19**

Your configuration file has been locked accidentally. What of the following command would you use to unlock?

- A. terraform filename-unlock
- B. state.tf-unlock
- C. terraform force-unlock
- D. delete the file and create a new state file

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 20**

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

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. If the state is changed, this may cause changes to occur during the next plan or apply.

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

### **NEW QUESTION: 21**

If a DevOps team adopts AWS Cloud Formation 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

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 22

You have modified your Terraform configuration to fix a typo in the Terraform ID of a resource from aws\_security\_group.http to aws\_security\_group.http

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"]
  }
}
```

Freecram.net

Updated configuration:

Which of the following commands would you run to update the ID in state without destroying the resource?

- A. terraform apply
- B. terraform refresh
- C. terraform mv aws-security-group.hpt aws-security-group.http

**Answer:** C ([LEAVE A REPLY](#))

### NEW QUESTION: 23

You have created a terraform script that uses a lot of new constructs that have been introduced in terraform v0.12. However, many developers who are cloning the script from your git repo, are using v0.11, and getting errors. What can be done from your end to solve this problem?

- A. Force developer to use v0.12 by using terraform setting 'required\_version' and set it to >=0.12.
- B. Refactor the code to support both v0.11, and v0.12. It might be a difficult process, but there is no other way.
- C. Add a condition in front of each such specific construct, to check whether the running terraform version id v0.11 or v0.12, and ,work accordingly.
- D. Add comments in your code to tell developers to use v0.12 . If they use v0.11 , that should be their problem , which they need to figure out.

**Answer:** ([SHOW ANSWER](#))

<https://www.terraform.io/docs/configuration/terraform.html>

### NEW QUESTION: 24

What advantage does an operations team that uses infrastructure as code have?

- A. The ability to reuse best practice configurations and settings
- B. The ability to delete infrastructure
- C. The ability to autoscale a group of servers

D. The ability to update existing infrastructure

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 25**

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. Terraform attempts to provision the resource up to three times before exiting with an error
- B. the terraform plan is rolled back and all provisioned resources are removed
- C. it is automatically deleted
- D. the resource is marked as tainted

**Answer:** ([SHOW ANSWER](#))

If a resource successfully creates but fails during provisioning, Terraform will error and mark the resource as "tainted". A resource that is tainted has been physically created, but can't be considered safe to use since provisioning failed. Terraform also does not automatically roll back and destroy the resource during the apply when the failure happens, because that would go against the execution plan: the execution plan would've said a resource will be created, but does not say it will ever be deleted.

### **NEW QUESTION: 26**

What is terraform refresh intended to detect?

- A. Terraform configuration code changes
- B. State file drift
- C. Empty state files
- D. Corrupt state files

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 27**

Which of the following connection types are supported by the remote-exec provisioner? (select two)

- A. WinRM
- B. UDP
- C. SMB
- D. RDP
- E. ssh

**Answer:** ([SHOW ANSWER](#))

The remote-exec provisioner invokes a script on a remote resource after it is created. The remote-exec provisioner supports both ssh and winrm type connections.

remote-exec connection types -

- \* ssh on Linux
- \* winrm on Windows

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

## **NEW QUESTION: 28**

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer:** ([SHOW ANSWER](#))

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

## **NEW QUESTION: 29**

Which of the below options is a valid interpolation syntax for retrieving a data source?

- A. \${google\_storage\_bucket.backend}
- B. \${azurerm\_resource\_group.test.data}
- C. \${aws\_instance.web.id.data}
- D. \${data.google\_dns\_keys.foo\_dns\_keys.key\_signing\_keys[0].ds\_record}

**Answer:** ([SHOW ANSWER](#))

Data source attributes are interpolated with the general syntax `data.TYPENAME.NAME.ATTRIBUTE`.

The interpolation for a resource is the same but without the `data.` prefix  
(`TYPENAME.NAME.ATTRIBUTE`).

<https://www.terraform.io/docs/configuration-0-11/interpolation.html#attributes-of-a-data-source>

## **NEW QUESTION: 30**

- A. Sentinel
- B. Cost Estimation
- C. Audit Logs
- D. Clustering
- E. SAML/SSO

**Answer:** ([SHOW ANSWER](#))

Sentinel and Cost Estimation are also available in Terraform Cloud

<https://www.hashicorp.com/products/terraform/pricing/>

## NEW QUESTION: 31

Module version is required to reference a module on the Terraform Module Registry.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

## NEW QUESTION: 32

You have been given requirements to create a security group for a new application. Since your organization standardizes on Terraform, you want to add this new security group with the fewest number of lines of code. What feature could you use to iterate over a list of required tcp ports to add to the new security group?

- A. dynamic backend
- B. splat expression
- C. terraform import
- D. dynamic block

**Answer:** ([SHOW ANSWER](#))

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.

<https://www.terraform.io/docs/configuration/expressions.html#dynamic-blocks>

## NEW QUESTION: 33

A user has created three workspaces using the command line - prod, dev, and test. The user wants to create a fourth workspace named stage. Which command will the user execute to accomplish this?

- A. terraform workspace new stage
- B. terraform workspace -new stage
- C. terraform workspace -create stage
- D. terraform workspace create stage

**Answer:** ([SHOW ANSWER](#))

The terraform workspace new command is used to create a new workspace.

<https://www.terraform.io/docs/commands/workspace/new.html>

## NEW QUESTION: 34

Given the Terraform configuration below, in which order will the resources be created?

```
1. resource "aws_instance" "web_server"  
2. {  
3.   ami = "ami-b374d5a5"  
4.   instance_type = "t2.micro"  
5. }  
6. resource "aws_eip" "web_server_ip"  
7. {  
8.   vpc = true  
9.   instance = aws_instance.web_server.id
```

- A. aws\_eip will be created first  
aws\_instance will be created second
- B. aws\_eip will be created first  
aws\_instance will be created second
- C. Resources will be created simultaneously
- D. aws\_instance will be created first  
aws\_eip will be created second

**Answer:** ([SHOW ANSWER](#))

Implicit and Explicit Dependencies

By studying the resource attributes used in interpolation expressions, Terraform can automatically infer when one resource depends on another. In the example above, the reference to `aws_instance.web_server.id` creates an implicit dependency on the `aws_instance` named `web_server`.

Terraform uses this dependency information to determine the correct order in which to create the different resources.

# Example of Implicit Dependency

```
resource "aws_instance" "web_server" {  
  ami = "ami-b374d5a5"  
  instance_type = "t2.micro"  
}  
resource "aws_eip" "web_server_ip" {  
  vpc = true  
  instance = aws_instance.web_server.id  
}
```

In the example above, Terraform knows that the `aws_instance` must be created before the `aws_eip`.

Implicit dependencies via interpolation expressions are the primary way to inform Terraform about these relationships, and should be used whenever possible.

Sometimes there are dependencies between resources that are not visible to Terraform. The depends\_on argument is accepted by any resource and accepts a list of resources to create explicit dependencies for.

For example, perhaps an application we will run on our EC2 instance expects to use a specific Amazon S3 bucket, but that dependency is configured inside the application code and thus not visible to Terraform. In that case, we can use depends\_on to explicitly declare the dependency:

```
# Example of Explicit Dependency
```

```
# New resource for the S3 bucket our application will use.
```

```
resource "aws_s3_bucket" "example" {
  bucket = "terraform-getting-started-guide"
  acl   = "private"
}

# Change the aws_instance we declared earlier to now include "depends_on" resource
"aws_instance" "example" { ami = "ami-2757f631" instance_type = "t2.micro"
# Tells Terraform that this EC2 instance must be created only after the
# S3 bucket has been created.
depends_on = [aws_s3_bucket.example]
}
```

<https://learn.hashicorp.com/terraform/getting-started/dependencies.html>

## NEW QUESTION: 35

You want to use different AMI images for different regions and for the purpose you have defined following code block.

```
1. variable "images"
2. {
3. type = "map"
4.
5. default = {
6. us-east-1 = "image-1234"
7. us-west-2 = "image-4567"
8. us-west-1 = "image-4589"
9. }
10. }
```

What of the following approaches needs to be followed in order to select image-4589?

- A. var.images["us-west-1"]
- B. var.images[3]
- C. lookup(var.images["us-west-1"])
- D. var.images[2]

**Answer: ([SHOW ANSWER](#))**

## NEW QUESTION: 36

By default, provisioners that fail will also cause the Terraform apply itself to error. How can you change this default behavior within a provisioner?

- A. provisioner "local-exec" { on\_failure = "next" }
- B. provisioner "local-exec" { when = "failure" terraform apply }
- C. provisioner "local-exec" { on\_failure = "continue" }
- D. provisioner "local-exec" { on\_failure = continue }

**Answer:** ([SHOW ANSWER](#))

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

### NEW QUESTION: 37

You have to initialize a Terraform backend before it can be configured.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 38

What is the standard workflow that a developer follows while working with terraform open source version?

- A. Run terraform refresh to update the terraform state , then write the terraform code , and finally run terraform apply.
- B. Run terraform destroy first since you need to start from fresh every time , before running terraform apply.
- C. Write terraform code , and run terraform push , to update the terraform state to the remote repo , which in turn will take care of the next steps.
- D. Write the terraform code on the developer machine , run terraform plan to check the changes , and run terraform apply to provision the infra.

**Answer:** ([SHOW ANSWER](#))

You do not need to run terraform refresh as terraform plan implicitly will run terraform refresh.

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

### NEW QUESTION: 39

If a module declares a variable with a default, that variable must also be defined within the module.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 40

What does this code do?

```
terraform {  
    required_providers {  
        aws = "≥ 3.0"  
    }  
}
```

**Freecram.net**

- A. Requires any version of the AWS provider after the 3.0 major release like 4.1
- B. Requires any version of the AWS provider > = 3.0 and < 4.0
- C. Requires any version of the AWS provider > 3.0
- D. Requires any version of the AWS provider > = 3.0

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 41**

Which of the following commands will launch the Interactive console for Terraform interpolations?

- A. `terraform console`
- B. `terraform cli`
- C. `terraform`
- D. `terraform cmdline`

**Answer:** B ([LEAVE A REPLY](#))

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

#### **NEW QUESTION: 42**

Anyone can publish and share modules on the Terraform Public Module Registry, and meeting the requirements for publishing a module is extremely easy. Select from the following list all valid requirements. (select three)

- A. The module must be PCI/HIPPA compliant.
- B. Module repositories must use this three-part name format, `terraform-- .`
- C. The registry uses tags to identify module versions.
- D. Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .
- E. The module must be on GitHub and must be a public repo.

**Answer:** C,D,E ([LEAVE A REPLY](#))

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

#### **NEW QUESTION: 43**

After creating a new workspace "PROD" you need to run the command `terraform select PROD` to switch to it.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

By default, when you create a new workspace you are automatically switched to it To create a new workspace and switch to it, you can use `terraform workspace new <new_workspace_name>`;

to switch to a existing workspace you can use terraform workspace select <existing\_workspace\_name>; Example:

```
$ terraform workspace new example
```

Created and switched to workspace "example"!

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state for this configuration.

#### **NEW QUESTION: 44**

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. Delete the Terraform VM resources from your Terraform code then run Terraform plan and terraform apply
- B. Use the terraform state rm command to remove the VM from state file
- C. Use the terraform apply command targeting the VM resources only
- D. Use the Terraform taint command targeting the VMs then run Terraform plan and Terraform apply

**Answer: ([SHOW ANSWER](#))**

#### **NEW QUESTION: 45**

What is the name assigned by Terraform to reference this resource?

```
resource "azurerm_resource_group" "dev" {  
    name = "test"  
    location = "westus"  
}
```

- A. azurerm\_resource\_group
- B. azurerm
- C. test
- D. dev

**Answer: ([SHOW ANSWER](#))**

#### **NEW QUESTION: 46**

Which of the following is not a valid Terraform collection type?

- A. set
- B. tree
- C. map
- D. list

**Answer: ([SHOW ANSWER](#))**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

#### **NEW QUESTION: 47**

The Security Operations team of ABC Enterprise wants to mandate that all the Terraform configuration that creates an S3 bucket must have encryption feature enabled. What is the best way to achieve it?

- A. Use Sentinel Policies.
- B. Use S3 bucket policy.
- C. Create a script that checks the encryption parameter is enabled on every git commit.
- D. Shared a SOP to engineers to mandate encryption feature on S3.

**Answer:** ([SHOW ANSWER](#))

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

Using Sentinel with Terraform Cloud involves:

- \* Defining the policies - Policies are defined using the policy language with imports for parsing the Terraform plan, state and configuration.
- \* Managing policies for organizations - Users with permission to manage policies can add policies to their organization by configuring VCS integration or uploading policy sets through the API. They also define which workspaces the policy sets are checked against during runs. (More about permissions.)
- \* 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.
- \* Mocking Sentinel Terraform data - Terraform Cloud provides the ability to generate mock data for any run within a workspace. This data can be used with the Sentinel CLI to test policies before deployment.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

#### **NEW QUESTION: 48**

What does terraform plan do ?

- A. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.
- B. Create an execution plan by evaluating the difference between configuration file and state file.
- C. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.

- D. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 49**

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.

**Answer:**

outFILENAME

#### **NEW QUESTION: 50**

A. True

B. False

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 51**

How can you trigger a run in a Terraform Cloud workspace that is connected to a Version Control System (VCS) repository?

- A. Only members of a VCS organization can open a pull request against repositories that are connected to Terraform Cloud workspaces
- B. Only Terraform Cloud organization owners can approve plans in VCS connected workspaces
- C. Only Terraform Cloud organization owners can set workspace variables on VCS connected workspaces
- D. Commit a change to the VCS working directory and branch that the Terraform Cloud workspace is connected to

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 52**

Which one of the following command will rewrite Terraform configuration files to a canonical format and style.

- A. `terraform graph -h`
- B. `terraform init`
- C. `terraform graph`
- D. `terraform fmt`

**Answer:** ([SHOW ANSWER](#))

The `terraform fmt` command is used to rewrite Terraform configuration files to a canonical format and style. This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability.

#### **NEW QUESTION: 53**

Why should secrets not be hard coded into Terraform code? Choose two correct answers

- A. It makes the code less reusable.
- B. The Terraform code is copied to the target resources to be applied locally and could expose secrets if a target resource is compromised.
- C. All passwords should be rotated on a quarterly basis.
- D. Terraform code is typically stored in version control, as well as copied to the systems from it's run. Any of those may not have robust security mechanisms.

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 54**

Select the most accurate statement to describe the Terraform language from the following list.

- A. Terraform is an immutable, declarative, Infrastructure as Code provisioning language based on Hashicorp Configuration Language, or optionally JSON.
- B. Terraform is a mutable, declarative, Infrastructure as Code configuration management language based on Hashicorp Configuration Language, or optionally JSON.
- C. Terraform is an immutable, procedural, Infrastructure as Code configuration management language based on Hashicorp Configuration Language, or optionally JSON.
- D. Terraform is a mutable, procedural, Infrastructure as Code provisioning language based on Hashicorp Configuration Language, or optionally YAML.

**Answer:** ([SHOW ANSWER](#))

Terraform is not a configuration management tool -

<https://www.terraform.io/intro/vs/chefpuppet.html> Terraform is a declarative language -

<https://www.terraform.io/docs/configuration/index.html> Terraform supports a syntax that is JSON compatible - <https://www.terraform.io/docs/configuration/syntax-json.html> Terraform is primarily designed on immutable infrastructure principles - <https://www.hashicorp.com/resources/what-is-mutable-vs-immutable-infrastructure>

#### **NEW QUESTION: 55**

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. 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
- B. Run `terraform destroy` then `terraform apply` and look for the IP address in `stdout`
- 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 output ip_address` to view the result

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 56**

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

**Answer:** ([SHOW ANSWER](#))

The terraform import command is used to import existing infrastructure. Import existing Google Cloud resources into Terraform with Terraformer.

### **NEW QUESTION: 57**

Jack is a newbie to Terraform and wants to enable detailed logging to find all the details. Which environment variable does he need to set?

- A. TF LOG
- B. TF\_var\_log
- C. TF\_help
- D. TF\_Debug

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 58**

You wanted to destroy some of the dependent resources from real infrastructure. You choose to delete those resources from your configuration file and run terraform plan and then apply. Which of the following way your resources would be destroyed?

- A. Terraform can still determine the correct order for destruction from the state even when you delete one or more items from the configuration.
- B. Those would be destroyed in the order in which they were written in the configuration file previously before you have deleted them from configuration file.
- C. The resource will be destructed in random order as you have already deleted them from configuration.
- D. You can not destroy resources by deleting them from configuration file and running plan and apply.

**Answer:** ([SHOW ANSWER](#))

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state. Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

#### **NEW QUESTION: 59**

Which backend does the Terraform CLI use by default?

- A. Remote
- B. Local
- C. Consul
- D. Terraform Cloud

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 60**

- A. Backend types
- B. Version control
- C. Provider constraints
- D. State locking

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 61**

True or False? When using the Terraform provider for Vault, the tight integration between these HashiCorp tools provides the ability to mask secrets in the terraform plan and state files.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

Currently, Terraform has no mechanism to redact or protect secrets that are returned via data sources, so secrets read via this provider will be persisted into the Terraform state, into any plan files, and in some cases in the console output produced while planning and applying. These artifacts must, therefore, all be protected accordingly.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (450 Q&As Dumps, **30%OFF Special Discount: freecram**)

#### **NEW QUESTION: 62**

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 to install the necessary plugins 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 you to format your code according to best practices first

Answer: ([SHOW ANSWER](#))

#### **NEW QUESTION: 63**

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

- A. True
- B. False

Answer: ([SHOW ANSWER](#))

#### **NEW QUESTION: 64**

In order to reduce the time it takes to provision resources, Terraform uses parallelism. By default, how many resources will Terraform provision concurrently?

- A. 20
- B. 10
- C. 5
- D. 50

Answer: ([SHOW ANSWER](#))

#### **NEW QUESTION: 65**

All standard backend types support state storage, locking, and remote operations like plan, apply and destroy.

- A. True
- B. False

Answer: ([SHOW ANSWER](#))

#### **NEW QUESTION: 66**

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. False
- B. True

Answer: ([SHOW ANSWER](#))

#### **NEW QUESTION: 67**

colleagues is new toTerraform and wants to add a new workspace named new-hire.

What command he should execute from the following?

- A. terraform workspace-new-new-hire
- B. terraform workspace init new-hire

C. `terraform workspace new new hire`

D. `terraform workspace new-hire`

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 68

You have created a custom variable definition file `my_vars.tfvars`. How will you use it for provisioning infrastructure?

A. `terraform apply -var-state-file ="my_vars.tfvars"`

B. `terraform apply var-file="my_vars.tfvars"`

C. `terraform plan -var-file="my_vars.tfvar"`

D. `terraform apply -var-file="my_vars.tfvars"`

**Answer:** ([SHOW ANSWER](#))

To set lots of variables, it is more convenient to specify their values in a variable definitions file (with a filename ending in either `.tfvars` or `.tfvars.json`) and then specify that file on the command line with `-var-file`:

`terraform apply -var-file="my_vars.tfvars"`

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

### NEW QUESTION: 69

You cannot publish your own modules on the Terraform Registry.

A. False

B. True

**Answer:** ([SHOW ANSWER](#))

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

### NEW QUESTION: 70

A. `TF_CLI_ARGS`

B. `TF_VAR`

C. `TF_VAR_`

D. `TF_VAR_ENV`

**Answer:** C ([LEAVE A REPLY](#))

Environment variables can be used to set variables. The environment variables must be in the format `TF_VAR_name` and this will be checked last for a value. For example:

`export TF_VAR_region=us-west-1`

`export TF_VAR_ami=ami-049d8641`

`export TF_VAR_alist='[1,2,3]'`

`export TF_VAR_amap='{ foo = "bar", baz = "qux" }'`

<https://www.terraform.io/docs/commands/environment-variables.html>

### NEW QUESTION: 71

What is the command you can use to set an environment variable named "var1" of type String?

- A. export TF\_VAR\_VAR1
- B. set TF\_VAR\_var1
- C. variable "var1" { type = "string"}
- D. export TF\_VAR\_var1

**Answer:** ([SHOW ANSWER](#))

The environment variable must be in the format TF\_VAR\_name, so for the

[https://www.terraform.io/docs/commands/environment-variables.html#tf\\_var\\_name](https://www.terraform.io/docs/commands/environment-variables.html#tf_var_name)

### NEW QUESTION: 72

Terraform variable names are saved in the state file.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 73

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. Git
- B. Amazon S3
- C. Artifactory
- D. Terraform Cloud

**Answer:** B ([LEAVE A REPLY](#))

### NEW QUESTION: 74

Your team has started using terraform OSS in a big way , and now wants to deploy multi region deployments (DR) in aws using the same terraform files . You want to deploy the same infra (VPC,EC2 ...) in both us-east-1 ,and us-west-2 using the same script , and then peer the VPCs across both the regions to enable DR traffic. But , when you run your script , all resources are getting created in only the default provider region. What should you do? Your provider setting is as below -

# The default provider configuration provider "aws" { region = "us-east-1" }

- A. No way to enable this via a single script . Write 2 different scripts with different default providers in the 2 scripts , one for us-east , another for us-west.
- B. Create a list of regions , and then use a for-each to iterate over the regions , and create the same resources ,one after the one , over the loop.
- C. Use provider alias functionality , and add another provider for us-west region . While creating the resources using the tf script , reference the appropriate provider (using the alias).
- D. Manually create the DR region , once the Primary has been created , since you are using terraform OSS , and multi region deployment is only available in Terraform Enterprise.

**Answer:** ([SHOW ANSWER](#))

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

<https://www.terraform.io/docs/configuration/providers.html>

## NEW QUESTION: 75

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 plan 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

### Answer: ([SHOW ANSWER](#))

changes, and terraform apply to create new infrastructure.

Reference:

+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new  
+infrastructure.&oq=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new+infrastructure.&aqs=chrome..69i57.556j0j7&sourceid=chrome&ie=UTF-8

## NEW QUESTION: 76

You have already set TF\_LOG = DEBUG to enable debug log. Now you want to always write the log to the directory you're currently running terraform from. what should you do to achieve this.

- A. Run the command `export TF_LOG_FILE=./terraform.log`.
- B. Run the command `export TF_LOG_PATH=./terraform.log`.
- C. Run the command `export TF_DEBUG_PATH=./terraform.log`.
- D. No explicit action required. Terraform will take care of this as you have enable `TF_LOG`.

**Answer: B ([LEAVE A REPLY](#))**

<https://www.terraform.io/docs/commands/environment-variables.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!  
Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the newest Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (450 Q&As Dumps, **30%OFF Special Discount: freecram**)

#### **NEW QUESTION: 77**

Named workspaces are not a suitable isolation mechanism for strong separation between staging and production?

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

#### **NEW QUESTION: 78**

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

**Answer: A ([LEAVE A REPLY](#))**

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

#### **NEW QUESTION: 79**

Multiple configurations for the same provider can be used in a single configuration file.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it creates an additional provider configuration. For providers that have no required configuration arguments, the implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-provider-instances>

### **NEW QUESTION: 80**

- A. Store the sensitive variables as plain text in a source code repository
- B. Pass variables to Terraform with a -var flag
- C. Copy the sensitive variables into your Terraform code
- D. Store the sensitive variables in a secure\_vars.tf file

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 81**

Which of the following is considered a Terraform plugin?

- A. Terraform language
- B. Terraform tooling
- C. Terraform logic
- D. Terraform provider

**Answer:** ([SHOW ANSWER](#))

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

<https://www.terraform.io/docs/plugins/basics.html>

### **NEW QUESTION: 82**

Which option can not be used to keep secrets out of Terraform configuration files?

- A. A Terraform provider
- B. A -var flag
- C. secure string
- D. Environment variables

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 83**

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

- A. terraform providers
- B. terraform state show
- C. terraform provider
- D. terraform plan

**Answer:** A ([LEAVE A REPLY](#))

Use the terraform providers command to view the specified version constraints for all providers used in the current configuration.

<https://www.terraform.io/docs/configuration/providers.html>

### **NEW QUESTION: 84**

Which of the following Terraform commands will automatically refresh the state unless supplied with additional flags or arguments? Choose TWO correct answers.

- A. terraform validate
- B. terraform state
- C. terraform plan
- D. terraform apply
- E. terraform output

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 85**

Resources in terraform can have same identifiers(Resource type + Block name).

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 86

```
1. resource "aws_s3_bucket" "example" {  
2.   bucket = "my-test-s3-terraform-bucket"  
3. }  
4. resource "aws_iam_role" "test_role" {  
5.   name = "test_role"  
6. }
```

Due to the way that the application code is written, the s3 bucket must be created before the test role is created, otherwise there will be a problem. How can you ensure that?

- A. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- B. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- C. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- D. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

**Answer:** ([SHOW ANSWER](#))

Implicit dependency works only if there is some reference of one resource to another. Explicit dependency is the option here.

## NEW QUESTION: 87

A single terraform resource file that defines an aws\_instance resource can simple be renamed to azurerm\_virtual\_machine in order to switch cloud providers

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Providers usually require some configuration of their own to specify endpoint URLs, regions, authentication settings.

Providers Initialization can be done by either explicitly via a provider block or by adding a resource from that provider

<https://www.terraform.io/docs/configuration/providers.html>

## NEW QUESTION: 88

Where does the Terraform local backend store its state?

- A. In the /tmp directory
- B. In the terraform.tfvars file
- C. In the terraform.tfstate file
- D. In the user's .terraformrc file

**Answer:** ([SHOW ANSWER](#))

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

## **NEW QUESTION: 89**

terraform refresh will update the state file?

A. True

B. False

**Answer: ([SHOW ANSWER](#))**

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. If the state is changed, this may cause changes to occur during the next plan or apply.

## **NEW QUESTION: 90**

A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.

B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.

C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.

D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer: ([SHOW ANSWER](#))**

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:

module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

aws\_instance.web[3]

Refers to only the last instance in the config, and an address like this:

aws\_instance.web

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

### NEW QUESTION: 91

Your manager has instructed you to start using terraform for the entire infra provisioning of the application stack. There are 4 environments - DEV , QA , UAT , and PROD. The application team has asked for complete segregation between these environments including the backend , state , and also configurations ,since there will be unique resources in different environments . What is the possible way to structure the terraform code to facilitate that.

- A. Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to a different backend.
- B. Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to the same backend.
- C. Implement terraform workspaces , and map each environment with one workspace.
- D. Enable remote backend storage . Configure 4 different backend storages , one for each environment.

#### Answer: ([SHOW ANSWER](#))

In particular, organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 92

If a module uses a local variable, you can expose that value with a terraform output.

- A. True
- B. False

#### Answer: ([SHOW ANSWER](#))

Output values are like function return values.

### NEW QUESTION: 93

Your company has been using Terraform Cloud for a some time now . But every team is creating their own modules , and there is no standardization of the modules , with each team creating the resources in their own unique way . You want to enforce a standardization of the modules across the enterprise . What should be your approach.

- A. Create individual workspaces for each team , and ask them to share modules across workspaces.
- B. Implement a Private module registry in Terraform cloud , and ask teams to reference them.
- C. Upgrade to Terraform enterprise , since this is not possible in terraform cloud.
- D. Upload the modules in the terraform public module registry , and ask teams to reference them

**Answer: ([SHOW ANSWER](#))**

Terraform Cloud's private module registry helps you share Terraform modules across your organization. It includes support for module versioning, a searchable and filterable list of available modules, and a configuration designer to help you build new workspaces faster.

By design, the private module registry works much like the public Terraform Registry. If you're already used the public registry, Terraform Cloud's registry will feel familiar.

Understand the different offerings in Terraform OS, Terraform Cloud and Terraform Enterprise. Terraform Cloud's private module registry helps you share Terraform modules across your organization.

<https://www.terraform.io/docs/cloud/registry/index.html>

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

#### **NEW QUESTION: 94**

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

**Answer: ([SHOW ANSWER](#))**

approval.

#### **NEW QUESTION: 95**

Given the Terraform configuration below, in which order will the resources be created?

- A. Larger image
- B. resources will be created simultaneously
- C. `aws_eip` will be created first `aws_instance` will be created second
- D. `aws_instance` will be created first `aws_eip` will be created second

**Answer: ([SHOW ANSWER](#))**

The aws\_instance will be created first, and then aws\_eip will be created second due to the aws\_eip's resource dependency of the aws\_instance id

### NEW QUESTION: 96

The terraform state command can be used to \_\_\_\_\_

- A. Update current state
- B. Refresh existing state file
- C. Print the current state file in console
- D. It is not a valid command

**Answer:** ([SHOW ANSWER](#))

The terraform state command is used for advanced state management. Rather than modify the state directly, the terraform state commands can be used in many cases instead.

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

### NEW QUESTION: 97

Which one is the right way to import a local module names consul?

- A. module "consul" { source = "consul"}
- B. module "consul" { source = "./consul"}
- C. module "consul" { source = "../consul"}
- D. module "consul" { source = "module/consul"}

**Answer:** ([SHOW ANSWER](#))

A local path must begin with either ./ or ../ to indicate that a local path is intended, to distinguish from a module registry address.

```
module "consul" {
  source = "./consul"
}
```

### NEW QUESTION: 98

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

You can change your backend configuration at any time. You can change both the configuration itself as well as the type of backend (for example from "consul" to "s3").

Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

## **NEW QUESTION: 99**

The terraform init 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.

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

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

## **NEW QUESTION: 100**

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

Sensitive output values are still recorded in the state, and so will be visible to anyone who is able to access the state data.

## **NEW QUESTION: 101**

What allows you to conveniently switch between multiple instances of a single configuration within its single backend?

- A. Local backends
- B. Providers
- C. Remote backends
- D. Workspaces

**Answer: ([SHOW ANSWER](#))**

Named workspaces allow conveniently switching between multiple instances of a single configuration within its single backend. ... A common use for multiple workspaces is to create a parallel, distinct copy of a set of infrastructure in order to test a set of changes before modifying the main production infrastructure.

Workspaces, allowing multiple states to be associated with a single configuration. The configuration still has only one backend, but multiple distinct instances of that configuration to be deployed without configuring a new backend or changing authentication credentials.

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

## **NEW QUESTION: 102**

Which of the below are paid features of Terraform Cloud?

- A. Full API Coverage
- B. Secure variable Storage
- C. Roles/ Team management
- D. Cost Estimation
- E. Private Module Registry
- F. Sentinel policies

**Answer: ([SHOW ANSWER](#))**

<https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 103**

Which task does terraform ini- not perform?

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

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 104**

When configuring a remote backend in Terraform, it might be a good idea to purposely omit some of the required arguments to ensure secrets and other important data aren't inadvertently shared with others. What are the ways the remaining configuration can be added to Terraform so it can initialize and communicate with the backend? (select three)

- A.** directly querying HashiCorp Vault for the secrets
- B.** command-line key/value pairs
- C.** use the -backend-config=PATH to specify a separate config file
- D.** interactively on the command line

**Answer: ([SHOW ANSWER](#))**

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process. There are several ways to supply the remaining arguments:

<https://www.terraform.io/docs/backends/init.html#backend-initialization>

### **NEW QUESTION: 105**

You are reviewing Terraform configurations for a big project in your company. You noticed that there are several identical sets of resources that appear in multiple configurations. What feature of Terraform would you recommend to use to reduce the amount of cloned configuration between the different configurations?

- A.** Packages
- B.** Backends
- C.** Provisioners
- D.** Modules

**Answer: ([SHOW ANSWER](#))**

Modules are reusable configuration packages that Terraform can share through a variety of sources including Terraform Registries, GitHub, and Amazon S3 buckets.

A module is a container for multiple resources that are used together. Modules can be used to create lightweight abstractions, so that you can describe your infrastructure in terms of its architecture, rather than directly in terms of physical objects.

Modules are reusable configuration packages that Terraform can share through a variety of sources including Terraform Registries, GitHub, and Amazon S3 buckets.

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

### **NEW QUESTION: 106**

You have created an AWS EC2 instance of type t2.micro through your terraform configuration file ec2.tf . Now you want to change the instance type from t2.micro to t2.medium. Accordingly you have changed your configuration file and ran terraform plan. After running terraform plan you check the output and saw one instance will be updated from t2.micro --> t2.medium. After this you went to grab a coffee without running terraform apply and meanwhile a member of your team changed the instance type of that EC2 instance to t2.medium from aws console. After coming to your desk you run terraform apply. What will happen?

- A. 1 resource will be updated and you will see the message : Apply Complete ! Resources : 0 added, 1 changed, 0 destroyed.
- B. terraform apply will throw an error.
- C. No resource will be updated and you will see the message : Apply Complete ! Resources : 0 added, 0 changed, 0 destroyed.
- D. The instance type will be changed to t2.micro and again will be changed to t2.medium

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### **NEW QUESTION: 107**

What is the purpose of using the local-exec provisioner? (Select Two)

- A. To invoke a local executable.
- B. Executes a command on the resource to invoke an update to the Terraform state.
- C. To execute one or more commands on the machine running Terraform.
- D. Ensures that the resource is only executed in the local infrastructure where Terraform is deployed.

**Answer:** ([SHOW ANSWER](#))

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.

Note that even though the resource will be fully created when the provisioner is run, there is no guarantee that it will be in an operable state - for example system services such as sshd may not be started yet on compute resources.

Example usage

```
resource "aws_instance" "web" {  
# ...  
provisioner "local-exec" {  
command = "echo ${aws_instance.web.private_ip} >> private_ips.txt"  
}  
}
```

Note: Provisioners should only be used as a last resort. For most common situations there are better alternatives.

<https://www.terraform.io/docs/provisioners/local-exec.html>

## NEW QUESTION: 108

Which of the following value will be accepted for my\_var?

1. variable "my\_var"

2. {

3. type = string

4. }

A. 15

B. "15"

C. Both A and B

D. None of the above

**Answer:** ([SHOW ANSWER](#))

The Terraform language will automatically convert number and bool values to string values when needed, and vice-versa as long as the string contains a valid representation of a number or boolean value.

Example

\* true converts to "true", and vice-versa

\* false converts to "false", and vice-versa

\* 15 converts to "15", and vice-versa

Where possible, Terraform automatically converts values from one type to another in order to produce the expected type. If this isn't possible, Terraform will produce a type mismatch error and you must update the configuration with a more suitable expression.

<https://www.terraform.io/docs/configuration/expressions.html#type-conversion>

## NEW QUESTION: 109

In order to make a Terraform configuration file dynamic and/or reusable, static values should be converted to use what?

- A. Input Parameters
- B. Module
- C. Regular Expressions
- D. Output Value

**Answer:** ([SHOW ANSWER](#))

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module's own source code, and allowing modules to be shared between different configurations.

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

#### **NEW QUESTION: 110**

- A. var.vpc\_cidrs[0]
- B. vpc\_cidrs["us-east-1"]
- C. var.vpc\_cidrs.0
- D. var.vpc\_cidrs["us-east-1"]

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 111**

Workspaces in Terraform provides similar functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

<https://www.terraform.io/docs/cloud/migrate/workspaces.html>

Workspaces, managed with the `terraform workspace` command, aren't the same thing as Terraform Cloud's workspaces. Terraform Cloud workspaces act more like completely separate working directories; CLI workspaces are just alternate state files.

#### **NEW QUESTION: 112**

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. False
- B. True

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 113**

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A. TF\_TRACE

- B. TF\_DEBUG
- C. TF\_LOG
- D. TF\_INFO

**Answer:** ([SHOW ANSWER](#))

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

#### **NEW QUESTION: 114**

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

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 115**

When TF\_LOG\_PATH is set, TF\_LOG must be set in order for any logging to be enabled.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

TF\_LOG\_PATH specifies where the log should persist its output to. Note that even when TF\_LOG\_PATH is set, TF\_LOG must be set in order for any logging to be enabled.

For example, to always write the log to the directory you're currently running terraform from:

```
export TF_LOG_PATH=./terraform.log  
export TF_LOG=TRACE
```

#### **NEW QUESTION: 116**

Your team uses terraform OSS . You have created a number of reusable modules for important , independent network components that you want to share with your team to enhance consistency . What is the correct option/way to do that?

- A. Terraform modules cannot be shared in OSS version . Each developer needs to maintain their own modules and leverage them in the main tf file.
- B. Upload your modules with proper versioning in the terraform public module registry . Terraform OSS is directly integrated with the public module registry , and can reference the modules from the code in the main tf file.
- C. Terraform module sharing is only available in Enterprise version via terraform private module registry , so no way to enable it in OSS version.

**D.** Store your modules in a NAS/ shared file server , and ask your team members to directly reference the code from there. This is the only viable option in terraform OSS ,which is better than individually maintaining module versions for every developer.

**Answer:** ([SHOW ANSWER](#))

Software development encourages code reuse through reusable artifacts, such as libraries, packages and modules. Most programming languages enable developers to package and publish these reusable components and make them available on a registry or feed. For example, Python has Python Package Index and PowerShell has PowerShell Gallery.

For Terraform users, the Terraform Registry enables the distribution of Terraform modules, which are reusable configurations. The Terraform Registry acts as a centralized repository for module sharing, making modules easier to discover and reuse.

The Registry is available in two variants:

- \* Public Registry houses official Terraform providers -- which are services that interact with an API to expose and manage a specific resource -- and community-contributed modules.
- \* Private Registry is available as part of the Terraform Cloud, and can host modules internally within an organization.

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

### **NEW QUESTION: 117**

Ric wants to enable detail logging and he wants highest verbosity of logs. Which of the following environment variable settings is correct option for him to select.

- A.** Set TF\_LOG = DEBUG
- B.** Set VAR\_TF = TRACE
- C.** Set TF\_LOG = TRACE
- D.** Set VAR\_TF\_LOG = TRACE

**Answer:** C ([LEAVE A REPLY](#))

<https://www.terraform.io/docs/internals/debugging.html>

### **NEW QUESTION: 118**

HashiCorp offers multiple versions of Terraform, including Terraform open-source, Terraform Cloud, and Terraform Enterprise. Which of the following Terraform features are only available in the Enterprise edition? (select four)

- A.** SAML/SSO
- B.** Sentinel
- C.** Audit Logs
- D.** Clustering
- E.** Private Module Registry
- F.** Private Network Connectivity

**Answer:** ([SHOW ANSWER](#))

While there are a ton of features that are available to open source users, many features that are part of the Enterprise offering are geared towards larger teams and enterprise functionality. To

see what specific features are part of Terraform Cloud and Terraform Enterprise, check out this link. <https://www.hashicorp.com/products/terraform/pricing/>

### NEW QUESTION: 119

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

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 120

- A. `terraform workspace`
- B. `terraform workspace show`
- C. `terraform workspace list`
- D. `terraform show workspace`

**Answer:** C ([LEAVE A REPLY](#))

`terraform workspace list`

The command will list all existing workspaces.

### NEW QUESTION: 121

You have provisioned some aws resources in your test environment through Terraform for a POC work. After the POC, now you want to destroy the resources but before destroying them you want to check what resources will be getting destroyed through terraform. what are the options of doing that? (Select TWO)

- A. Use `terraform destroy` command
- B. This is not possible
- C. Use `terraform plan` command
- D. Use `terraform plan -destroy` command.

**Answer:** ([SHOW ANSWER](#))

<https://learn.hashicorp.com/terraform/getting-started/destroy>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest**

Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (450 Q&As Dumps, 30%OFF Special Discount: freecram)

### NEW QUESTION: 122

How is terraform import run?

- A. By an explicit call
- B. As a part of terraform refresh
- C. As a part of terraform plan
- D. All of the above
- E. As a part of terraform init

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 123

Which of the following statements about local modules is incorrect:

- A. Local modules are sourced from a directory on disk
- B. Local modules are not cached by terraform init command
- C. None of the above (all statements above are correct)
- D. Local modules support versions
- E. All of the above (all statements above are incorrect)

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 124

Which of the following challenges would Terraform be a candidate for solving? (Select THREE)

- A. Enable self-service infrastructure to allocate resources on your proprietary private cloud.
- B. Utilize a single tool for all of the infrastructure and configuration management needs.
- C. Have a single interoperable tool to manage the variety of services including GitHub repositories, MySQL database, and Kubernetes clusters.
- D. Reduce the number of workflows needed for managing infrastructure across each of the companies public and private clouds.

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 125

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

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

Answer: ([SHOW ANSWER](#))

## NEW QUESTION: 126

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`

**Answer:** B ([LEAVE A REPLY](#))

## NEW QUESTION: 127

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. `aws_instance.web[1].name`
- B. `aws_instance.web[2].name`
- C. `aws_instance.web[1]`
- D. `element(aws_instance.web, 2)`
- E. `aws_instance.web.*.name`

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 128

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 state`
- B. `terraform init`
- C. `terraform import`
- D. `terraform workspace`

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 129

Matt wants to import a manually created EC2 instance into terraform so that he can manage the EC2 instance through terraform going forward. He has written the configuration file of the EC2 instance before importing it to Terraform. Following is the code:

```
resource "aws_instance" "matt_ec2" { ami = "ami-bg2640de" instance_type = "t2.micro"  
vpc_security_group_ids = ["sg-6ae7d613", "sg-53370035"] key_name = "mysecret" subnet_id =  
"subnet-9e3cfbc5" } The instance id of that EC2 instance is i-0260835eb7e9bd40 How he can  
import data of EC2 to state file?
```

- A. terraform import aws\_instance.id = i-0260835eb7e9bd40
- B. terraform import i-0260835eb7e9bd40
- C. terraform import aws\_instance.i-0260835eb7e9bd40
- D. terraform import aws\_instance.matt\_ec2 i-0260835eb7e9bd40

**Answer: D ([LEAVE A REPLY](#))**

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

### NEW QUESTION: 130

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

### NEW QUESTION: 131

Which of the below options is the equivalent Terraform 0.12 version of the snippet which is written in Terraform 0.11?

- "\${var.instance\_id}"
- A. var.instance\_ids
- B. var.instance\_id
- C. None of the above
- D. variable.instance\_id

**Answer: ([SHOW ANSWER](#))**

### NEW QUESTION: 132

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"  
  }  
}
```

Freeecram.net

You immediately run `terraform apply` but don't see any changes. Your state file didn't move.  
Which command will migrate your current state file to the new S3 remote backend?

- A. `terraform refresh`
  - B. `terraform push`
  - C. `terraform init`
  - D. `terraform state`
- Answer: ([SHOW ANSWER](#))**

### NEW QUESTION: 133

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. Nothing, provisioners will not show errors in the command line
- C. The resource will be marked as "tainted"
- D. The resource will be destroyed

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 134**

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. Delete the Terraform state file and execute Terraform apply.
- B. The Terraform state file only contains the one new VM. Execute terraform destroy.
- C. The Terraform state file contains all 16 VMs in the team account. Execute terraform destroy and select the newly-created VM.
- D. Delete the VM using the cloud provider console and terraform apply to apply the changes to the Terraform state file.

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 135**

Why is it a good idea to declare the required version of a provider in a Terraform configuration file?

```
1. terraform
2. {
3.   required_providers
4. {
5.   aws = "~> 1.0"
6. }
7. }
```

- A. To ensure that the provider version matches the version of Terraform you are using.
- B. To remove older versions of the provider.
- C. Providers are released on a separate schedule from Terraform itself; therefore a newer version could introduce breaking changes.
- D. To match the version number of your application being deployed via Terraform.

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 136**

Your company has a lot of workloads in AWS , and Azure that were respectively created using CloudFormation , and AzureRM Templates. However , now your CIO has decided to use

Terraform for all new projects , and has asked you to check how to integrate the existing environment with terraform code. What should be your next plan of action?

- A. This is only possible in Terraform Enterprise , which has the TerraformConverter exe that can take any other template language like AzureRM and convert to Terraform code.
- B. Use terraform import command to import each resource one by one .
- C. Tell the CIO that this is not possible . Resources created in CloudFormation , and AzureRM templates cannot be tracked using terraform.
- D. Just write the terraform config file for the new resources , and run terraform apply , the state file will automatically be updated with the details of the new resources to be imported.

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

#### **NEW QUESTION: 137**

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 two spaces
- B. With three spaces
- C. With a tab
- D. With four spaces

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

# HashiCorp.TA-002-P.v2023-01-06.q165

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	165
<b>Version:</b>	v2023-01-06
<b># of views:</b>	102
<b># of Questions views:</b>	1790

<https://www.freecram.net/torrent/HashiCorp.TA-002-P.v2023-01-06.q165.html>

## NEW QUESTION: 1

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.

**Answer:** ([SHOW ANSWER](#))

Explanation

You develop a Terraform configuration containing one VM, perform terraform apply, and see that your VM

was created successfully. read the question carefully "Terraform configuration containing one VM, perform

terraform apply" so only one VM is in state file.

## NEW QUESTION: 2

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

**Answer:** B ([LEAVE A REPLY](#))

Explanation

The variable declaration can also include a default argument.

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

### **NEW QUESTION: 3**

What does Terraform use .terraform.lock.hcl file for?

- A.** Tracking provider dependencies Most Voted
- B.** There is no such file
- C.** Preventing Terraform runs from occurring
- D.** Storing references to workspaces which are locked

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/language/files/dependency-lock>

"hcl , and this name is intended to signify that it is a lock file for various items that Terraform caches in the .

terraform subdirectory of your working directory. Terraform automatically creates or updates the dependency

lock file each time you run the terraform init command."

### **NEW QUESTION: 4**

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

**Answer:** B ([LEAVE A REPLY](#))

Explanation

"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>

#### **NEW QUESTION: 5**

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

- A. terraform providers
- B. terraform state show
- C. terraform provider
- D. terraform plan

**Answer:** ([SHOW ANSWER](#))

Explanation

Use the terraform providers command to view the specified version constraints for all providers used in the current configuration.

<https://www.terraform.io/docs/configuration/providers.html>

#### **NEW QUESTION: 6**

What does terraform refresh command do?

- A. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- B. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- C. terraform refresh command basically updates the configuration file with the current state of the actual infrastructure
- D. terraform refresh syncs the state file with the real world infrastructure.
- E. terraform refresh is used to change/modify the infrastructure based on the existing state file, at that moment.

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 7**

terraform validate validates the syntax of Terraform files.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

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

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

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

## NEW QUESTION: 8

Terraform will sync all resources in state by default for every plan and apply, hence for larger infrastructures

this can slow down terraform plan and terraform apply commands?

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

Explanation

For small infrastructures, Terraform can query your providers and sync the latest attributes from all your

resources. This is the default behavior of Terraform: for every plan and apply, Terraform will sync all

resources in your state.

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to

query multiple resources at once, and the round trip time for each resource is hundreds of milliseconds. On top

of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number

of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as

the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

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

## NEW QUESTION: 9

Taint the resource "aws\_instance" "baz" resource that lives in module bar which lives in module foo.

- A. `terraform taint module.foo.module.bar.baz`
- B. `terraform taint module.foo.bar.aws_instance.baz`
- C. `terraform taint module.foo.module.bar.aws_instance.baz`
- D. `terraform taint foo.bar.aws_instance.baz`

**Answer: ([SHOW ANSWER](#))**

Explanation

Check resource addressing

<https://www.terraform.io/docs/internals/resource-addressing.html>

### **NEW QUESTION: 10**

Resources in terraform can have same identifiers(Resource type + Block name).

A. False

B. True

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 11**

You want terraform plan and terraform apply to be executed in Terraform Cloud's run environment but the

output is to be streamed locally. Which one of the below you will choose?

A. Local Backends.

B. Terraform Backends.

C. This can be done using any of the local or remote backends.

D. Remote Backends.

**Answer:** ([SHOW ANSWER](#))

Explanation

When using full remote operations, operations like terraform plan or terraform apply can be executed in

Terraform Cloud's run environment, with log output streaming to the local terminal. Remote plans and applies

use variable values from the associated Terraform Cloud workspace.

Terraform Cloud can also be used with local operations, in which case only state is stored in the Terraform

Cloud backend.

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

### **NEW QUESTION: 12**

terraform destroy is the only way to remove infrastructure.

A. True

B. False

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 13**

Which backend does the Terraform CLI use by default?

A. Terraform Cloud

B. Consul

C. Remote

D. Local

**Answer: (SHOW ANSWER)**

Explanation

"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>

#### **NEW QUESTION: 14**

A terraform apply can not \_\_\_\_\_ infrastructure.

- A. change
- B. destroy
- C. provision
- D. import

**Answer: (SHOW ANSWER)**

Explanation

<https://www.educative.io/answers/what-is-the-command-to-destroy-infrastructure-in-terraform>

#### **NEW QUESTION: 15**

What are the benefits of using Infrastructure as Code? (select five)

- A. Infrastructure as Code is relatively simple to learn and write, regardless of a user's prior experience with developing code
- B. Infrastructure as Code provides configuration consistency and standardization among deployments
- C. Infrastructure as Code is easily repeatable, allowing the user to reuse code to deploy similar, yet different resources
- D. Infrastructure as Code gives the user the ability to recreate an application's infrastructure for disaster recovery scenarios
- E. Infrastructure as Code easily replaces development languages such as Go and .Net for application development
- F. Infrastructure as Code allows a user to turn a manual task into a simple, automated deployment (Correct)

**Answer: (SHOW ANSWER)**

Explanation

If you are new to infrastructure as code as a concept, it is the process of managing infrastructure in a file or files rather than manually configuring resources in a user interface. A resource in this instance is any piece of infrastructure in a given environment, such as a virtual machine, security group, network interface, etc. At a high level, Terraform allows operators to use HCL to author files containing definitions of their desired resources on almost any provider (AWS, GCP, GitHub, Docker, etc) and automates the creation of those resources at the time of application.

### NEW QUESTION: 16

You have created a custom variable definition file testing.tfvars. How will you use it for provisioning infrastructure?

- A. terraform apply -var-state-file ="testing.tfvars"
- B. terraform plan -var-file="testing.tfvar"
- C. terraform apply -var-file="testing.tfvars"
- D. terraform apply var-file="testing.tfvars"

**Answer:** ([SHOW ANSWER](#))

Explanation

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 17

Why should secrets not be hard coded into Terraform code? Choose two correct answers

- A. All passwords should be rotated on a quarterly basis.
- B. Terraform code is typically stored in version control, as well as copied to the systems from h it's run.

Any of those may not have robust security mechanisms.

- C. It makes the code less reusable.
- D. The Terraform code is copied to the target resources to be applied locally and could expose secrets if a target resource is compromised.

**Answer: B,D** ([LEAVE A REPLY](#))

**NEW QUESTION: 18**

Using the terraform state rm command against a resource will destroy it.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

**NEW QUESTION: 19**

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

**Answer:** ([SHOW ANSWER](#))

Explanation

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>

**NEW QUESTION: 20**

Which of the following best describes the default local backend?

- A. The local backend is where Terraform Enterprise stores logs to be processed by an log collector.
- B. The local backend stores state on the local filesystem, locks the state using system APIs, and performs operations locally.
- C. The local backend is the directory where resources deployed by Terraform have direct access to in order to update their current state.
- D. The local backend is how Terraform connects to public cloud services, such as AWS, Azure, or GCP.

**Answer:** ([SHOW ANSWER](#))

Explanation

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

operations locally.

terraform {

backend "local" {

```
path = "relative/path/to/terraform.tfstate"
}
}

https://www.terraform.io/docs/backends/types/local.html
```

### NEW QUESTION: 21

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

**Answer:** ([SHOW ANSWER](#))

Explanation

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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>

### NEW QUESTION: 22

When should you use the force-unlock command?

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

**Answer:** ([SHOW ANSWER](#))

Explanation

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

### NEW QUESTION: 23

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 Most Voted
- D. The module's Terraform code

**Answer:** ([SHOW ANSWER](#))

Explanation

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

#### **NEW QUESTION: 24**

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 ]

**Answer:** ([SHOW ANSWER](#))

Explanation

<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.

#### **NEW QUESTION: 25**

Which of the below datatype is not supported by Terraform.

- A. Map
- B. Object
- C. Array
- D. List

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 26**

Terraform import command can import resources into modules as well directly into the root of your state.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

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 directly into the root of your state.

Terraform is able to import existing infrastructure. This allows us to take resources we've created by some other means (i.e. via console) and bring it under Terraform management.

This is a great way to slowly transition infrastructure to Terraform.

The `terraform import` command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it

will be known to Terraform. For example:

```
resource "aws_instance" "import_example" {  
  # ...instance configuration...  
}
```

Now `terraform import` can be run to attach an existing instance to this resource configuration:

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f)  
Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID `i-03efafa258104165f` (which has been created outside

Terraform) and attaches its existing settings, as described by the EC2 API, to the name `aws_instance.import_example` in the Terraform state.

As a result of the above command, the resource is recorded in the state file. We can now run `terraform plan` to

see how the configuration compares to the imported resource, and make any adjustments to the configuration

to align with the current (or desired) state of the imported object.

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

## NEW QUESTION: 27

After creating a new workspace "PROD" you need to run the command `terraform select PROD` to switch to it.

A. False

B. True

**Answer: ([SHOW ANSWER](#))**

Explanation

By default, when you create a new workspace you are automatically switched to it

To create a new workspace and switch to it, you can use `terraform workspace new <new_workspace_name>`

to switch to a existing workspace you can use terraform workspace select <existing\_workspace\_name>;

Example:

```
$ terraform workspace new example
```

Created and switched to workspace "example"!

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan"

Terraform will not see any existing state for this configuration.

### NEW QUESTION: 28

Which of the following can you do with terraform plan? Choose two correct answers.

- A. View the execution plan and check if the changes match your expectations
- B. Schedule Terraform to run at a planned time in the future
- C. Execute a plan in a different workspace
- D. Save a generated execution plan to apply later

Answer: ([SHOW ANSWER](#))

Explanation

<https://learn.hashicorp.com/tutorials/terraform/plan>

### NEW QUESTION: 29

Why might a user opt to include the following snippet in their configuration file?

- A. The user wants to ensure that the application being deployed is a minimum version of 0.12
- B. Terraform 0.12 introduced substantial changes to the syntax used to write Terraform configuration
- C. this ensures that all Terraform providers are above a certain version to match the application being deployed
- D. versions before Terraform 0.12 were not approved by HashiCorp to be used in production

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 30

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  
}
```

A. Option B

B. Option A

**Answer:** ([SHOW ANSWER](#))

#### NEW QUESTION: 31

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

A. False

B. True

**Answer:** ([SHOW ANSWER](#))

Explanation

You can change your backend configuration at any time. You can change both the configuration itself as well

as the type of backend (for example from "consul" to "s3").

Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of

the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

#### NEW QUESTION: 32

Suppose terraformcode is taking up some values which are not defined inside the code files. In which of the following options issue might have occurred?

- A. Issue in vars.tf file
- B. Issue in terraform.tfvars
- C. Issue in main.tf file
- D. Issue in Environment Variables

**Answer:** D ([LEAVE A REPLY](#))

### **NEW QUESTION: 33**

Which one of the following command will rewrite Terraform configuration files to a canonical format and style.

- A. terraform graph -h
- B. terraform init
- C. terraform graph
- D. terraform fmt

**Answer:** ([SHOW ANSWER](#))

Explanation

The terraform fmt command is used to rewrite Terraform configuration files to a canonical format and style.

This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability.

### **NEW QUESTION: 34**

What does terraform import allow you to do?

- A. Import a new Terraform module
- B. Import provisioned infrastructure to your state file
- C. Use a state file to import infrastructure to the cloud
- D. Import an existing state file to a new Terraform workspace

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 35**

What is the purpose of using the local-exec provisioner? (Select Two)

- A. To invoke a local executable.
- B. Executes a command on the resource to invoke an update to the Terraform state.
- C. To execute one or more commands on the machine running Terraform.
- D. Ensures that the resource is only executed in the local infrastructure where Terraform is deployed.

**Answer:** ([SHOW ANSWER](#))

## Explanation

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.

Note that even though the resource will be fully created when the provisioner is run, there is no guarantee that it will be in an operable state - for example system services such as sshd may not be started yet on compute resources.

## Example usage

```
resource "aws_instance" "web" {  
  # ...  
  provisioner "local-exec" {  
    command = "echo ${aws_instance.web.private_ip} >> private_ips.txt"  
  }  
}
```

Note: Provisioners should only be used as a last resort. For most common situations there are better alternatives.

<https://www.terraform.io/docs/provisioners/local-exec.html>

## NEW QUESTION: 36

To check if all code in a Terraform configuration with multiple modules is properly formatted without making changes, what command should be run?

- A. `terraform fmt -check`
- B. `terraform fmt -write=false`
- C. `terraform fmt "list -recursive`
- D. `terraform fmt -check -recursive`

**Answer:** ([SHOW ANSWER](#))

## Explanation

`-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.

## NEW QUESTION: 37

Your developers are facing a lot of problem while writing complex expressions involving difficult interpolations . They have to run the terraform plan every time and check whether there are errors , and also

check terraform apply to print the value as a temporary output for debugging purposes. What should be done to avoid this?

- A. Use terraform console command to have an interactive UI with full access to the underlying terraform state to run your interpolations , and debug at real-time.
- B. Add a breakpoint in your code, using the watch keyword , and output the value to console for temporary debugging.
- C. Use terraform zipmap function , it will be able to easily do the interpolations without complex code.
- D. Use terraform console command to have an interactive UI , but you can only use it with local state , and it does not work with remote state.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The terraform console command provides an interactive console for evaluating expressions. This is useful for

testing interpolations before using them in configurations, and for interacting with any values currently saved in state.

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

### **NEW QUESTION: 38**

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.

**Answer:**

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.

### **NEW QUESTION: 39**

Any user can publish modules to the public Terraform Module Registry.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 40**

What is the name assigned by Terraform to reference this resource?

```
mainresource "google_compute_instance" "main" {  
    name = "test"  
}
```

- A. main
- B. google
- C. test
- D. compute\_instance

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 41**

Terraform plan updates your state file.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

The terraform plan command creates an execution plan, which lets you preview the changes that Terraform

plans to make to your infrastructure. 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. Source:  
<https://www.terraform.io/cli/commands/plan>

### **NEW QUESTION: 42**

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  
  }  
}
```

**A. Option A**

- B. Option B
- C. Option C

**Answer:** ([SHOW ANSWER](#))

Explanation

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

#### **NEW QUESTION: 43**

Select the answer below that completes the following statement: Terraform Cloud can be managed from the

CLI but requires \_\_\_\_\_?

- A. an API token
- B. a TOTP token
- C. a username and password
- D. authentication using MFA

**Answer:** ([SHOW ANSWER](#))

Explanation

API and CLI access are managed with API tokens, which can be generated in the Terraform Cloud UI. Each

user can generate any number of personal API tokens, which allow access with their own identity and

permissions. Organizations and teams can also generate tokens for automating tasks that aren't tied to an individual user.

#### **NEW QUESTION: 44**

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 repository on GitHub
- C. Public Terraform Module Registry
- D. Private repository on GitHub

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 45**

When using remote state, state is only ever held in memory when used by Terraform.

- A. False
- B. True

**Answer:** B ([LEAVE A REPLY](#))

#### **NEW QUESTION: 46**

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\_10G\_PATM
- B. TF\_LOG
- C. TF\_10G\_LEVEL
- D. TF.LOG.FUE

**Answer:** B ([LEAVE A REPLY](#))

Explanation

<https://www.terraform.io/internals/debugging>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the newest Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (450 Q&As Dumps, **30%OFF Special Discount: freecram**)

#### NEW QUESTION: 47

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 Most Voted
- 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

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/language/modules/sources#selecting-a-revision>

#### NEW QUESTION: 48

By default, where does Terraform store its state file?

- A. Amazon S3 bucket
- B. shared directory
- C. remotely using Terraform Cloud
- D. current working directory

**Answer:** ([SHOW ANSWER](#))

## Explanation

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

## NEW QUESTION: 49

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

**Answer: B ([LEAVE A REPLY](#))**

## Explanation

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>

## NEW QUESTION: 50

Refer to the below code where developer is outputting the value of the database password but has used sensitive parameter to hide the output value in the CLI.

```
output "db_password" { value = aws_db_instance.db.password description = "The password for logging in to the database." sensitive = true}
```

Since sensitive is set to true, the value associated with db password will not be present in state file as plain-text?

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

## Explanation

Sensitive output values are still recorded in the state, and so will be visible to anyone who is able to access the state data.

## NEW QUESTION: 51

You want to use different AMI images for different regions and for the purpose you have defined following code block.

- 1.variable "images"
- 2.{
3. type = "map"
- 4.

```
5. default = {  
6.   us-east-1 = "image-1234"  
7.   us-west-2 = "image-4567"  
8.   us-west-1 = "image-4589"  
9. }  
10.}
```

What of the following approaches needs to be followed in order to select image-4589?

- A. var.images[2]
- B. var.images["us-west-1"]
- C. var.images[3]
- D. lookup(var.images["us-west-1"])

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 52

The terraform init 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.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

Explanation

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

### NEW QUESTION: 53

You have created a custom variable definition file `my_vars.tfvars`. How will you use it for provisioning infrastructure?

- A. `terraform apply -var-state-file ="my_vars.tfvars"`
- B. `terraform apply var-file="my_vars.tfvars"`
- C. `terraform plan -var-file="my_vars.tfvar"`
- D. `terraform apply -var-file="my_vars.tfvars"`

**Answer:** ([SHOW ANSWER](#))

Explanation

To set lots of variables, it is more convenient to specify their values in a variable definitions file (with a filename ending in either `.tfvars` or `.tfvars.json`) and then specify that file on the command line with `-var-file`:

`terraform apply -var-file="my_vars.tfvars"`

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

## **NEW QUESTION: 54**

Terraform Enterprise (also referred to as pTFE) requires what type of backend database for a clustered deployment?

- A.** PostgreSQL
- B.** Cassandra
- C.** MySQL
- D.** MSSQL

**Answer:** ([SHOW ANSWER](#))

Explanation

External Services mode stores the majority of the stateful data used by the instance in an external PostgreSQL

database and an external S3-compatible endpoint or Azure blob storage. There is still critical data stored on the

instance that must be managed with snapshots. Be sure to check the PostgreSQL Requirements for information

that needs to be present for Terraform Enterprise to work. This option is best for users with expertise

managing PostgreSQL or users that have access to managed PostgreSQL offerings like AWS RDS.

## **NEW QUESTION: 55**

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.** Reinitialize because your configuration has changed
- B.** Run an apply and confirm the planned changes
- C.** Inspect your Terraform state because you want to change it
- D.** Inspect all Terraform outputs to make sure they are correct

**Answer:** ([SHOW ANSWER](#))

## **NEW QUESTION: 56**

In the below configuration, how would you reference the module output vpc\_id?

```
module "vpc" {
  source = "terraform-and-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.

**Answer:**

module.vpc.vpc\_id  
<https://cloudcasts.io/course/terraform/community-vpc-module>

**NEW QUESTION: 57**

Which of the following challenges would Terraform be a candidate for solving? (Select THREE)

- A. Enable self-service infrastructure to allocate resources on your proprietary private cloud.
- B. Reduce the number of workflows needed for managing infrastructure across each of the companies public and private clouds.
- C. Have a single interoperable tool to manage the variety of services including GitHub repositories, MySQL database, and Kubernetes clusters.
- D. Utilize a single tool for all of the infrastructure and configuration management needs.

**Answer:** ([SHOW ANSWER](#))

**NEW QUESTION: 58**

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

**Answer:** ([SHOW ANSWER](#))

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

**NEW QUESTION: 59**

Module version is required to reference a module on the Terraform Module Registry.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 60

```
resource "aws_s3_bucket" "example" { bucket = "my-test-s3-terraform-bucket" ...} resource  
"aws_iam_role"
```

```
"test_role" { name = "test_role" ...}
```

Due to the way that the application code is written , the s3 bucket must be created before the test role is created , otherwise there will be a problem. How can you ensure that?

- A. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- B. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- C. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.
- D. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.

**Answer:** ([SHOW ANSWER](#))

Explanation

Use the depends\_on meta-argument to handle hidden resource dependencies that Terraform can't automatically infer.

Explicitly specifying a dependency is only necessary when a resource relies on some other resource's behavior but doesn't access any of that resource's data in its arguments.

### NEW QUESTION: 61

By default, provisioners that fail will also cause the Terraform apply itself to error. How can you change this

default behavior within a provisioner?

- A. provisioner "local-exec" { on\_failure = "next" }
- B. provisioner "local-exec" { when = "failure" terraform apply }
- C. provisioner "local-exec" { on\_failure = "continue" }
- D. provisioner "local-exec" { on\_failure = continue }

**Answer:** ([SHOW ANSWER](#))

Explanation

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 62

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

**Answer:** ([SHOW ANSWER](#))

Explanation

<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.

### NEW QUESTION: 63

In the example below, the depends\_on argument creates what type of dependency?

- A. explicit dependency
- B. implicit dependency
- C. internal dependency
- D. non-dependency resource

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 64

1. resource "aws\_s3\_bucket" "example" {
2. bucket = "my-test-s3-terraform-bucket"
3. ...} resource "aws\_iam\_role" "test\_role" {
4. name = "test\_role"
5. ...}

Due to the way that the application code is written, the s3 bucket must be created before the test role is

created, otherwise there will be a problem. How can you ensure that?

- A.** Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- B.** This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- C.** This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- D.** Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

**Answer:** ([SHOW ANSWER](#))

Explanation

Implicit dependency works only if there is some reference of one resource to another. Explicit dependency is the option here.

## NEW QUESTION: 65

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

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://docs.hashicorp.com/sentinel/intro/what>

<https://medium.com/hashicorp-engineering/enforcing-aws-s3-security-best-practice-using-terraform-sentinel-ddc>

## NEW QUESTION: 66

Your team uses terraform OSS . You have created a number of reusable modules for important , independent network components that you want to share with your team to enhance consistency . What is the correct option/way to do that?

- A.** Terraform modules cannot be shared in OSS version . Each developer needs to maintain their own modules and leverage them in the main tf file.

**B.** Upload your modules with proper versioning in the terraform public module registry . Terraform OSS is

directly integrated with the public module registry , and can reference the modules from the code in the

main tf file.

**C.** Terraform module sharing is only available in Enterprise version via terraform private module registry ,

so no way to enable it in OSS version.

**D.** Store your modules in a NAS/ shared file server , and ask your team members to directly reference the

code from there. This is the only viable option in terraform OSS ,which is better than individually maintaining module versions for every developer.

**Answer:** ([SHOW ANSWER](#))

Explanation

Software development encourages code reuse through reusable artifacts, such as libraries, packages and

modules. Most programming languages enable developers to package and publish these reusable components

and make them available on a registry or feed. For example, Python has Python Package Index and PowerShell

has PowerShell Gallery.

For Terraform users, the Terraform Registry enables the distribution of Terraform modules, which are reusable

configurations. The Terraform Registry acts as a centralized repository for module sharing, making modules

easier to discover and reuse.

The Registry is available in two variants:

\* Public Registry houses official Terraform providers -- which are services that interact with an API to expose

and manage a specific resource -- and community-contributed modules.

\* Private Registry is available as part of the Terraform Cloud, and can host modules internally within an organization.

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

## NEW QUESTION: 67

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

**Answer: ([SHOW ANSWER](#))**

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

### **NEW QUESTION: 68**

In the following code snippet, the block type is identified by which string?

A. resource

B. instance\_type

C. "aws\_instance"

D. "db"

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 69**

terraform state subcommands such as list are read-only commands, do read-only commands  
create state  
backup files?

A. Yes

B. No

**Answer: ([SHOW ANSWER](#))**

Explanation

Subcommands that are read-only (such as list) do not write any backup files since they aren't  
modifying the  
state.

All terraform state subcommands that modify the state write backup files. The path of these  
backup file can be  
controlled with -backup.

<https://www.terraform.io/docs/commands/state/index.html#backups>

### **NEW QUESTION: 70**

You have declared a variable name my\_var in terraform configuration without a value associated with it.  
variable my\_var {}

After running terraform plan it will show an error as variable is not defined.

A. True

B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

Input variables are usually defined by stating a name, type and a default value. However, the type  
and default  
values are not strictly necessary. Terraform can deduct the type of the variable from the default or  
input value.

Variables can be predetermined in a file or included in the command-line options. As such, the simplest

variable is just a name while the type and value are selected based on the input.

```
variable "variable_name" {}
```

```
terraform apply -var variable_name="value"
```

The input variables, like the one above, use a couple of different types: strings, lists, maps, and boolean. Here

are some examples of how each type are defined and used.

String

Strings mark a single value per structure and are commonly used to simplify and make complicated values

more user-friendly. Below is an example of a string variable definition.

```
variable "template" {
  type = string
  default = "01000000-0000-4000-8000-000030080200"
}
```

A string variable can then be used in resource plans. Surrounded by double quotes, string variables are a

simple substitution such as the example underneath.

```
storage = var.template
```

List

Another type of Terraform variables lists. They work much like a numbered catalogue of values.

Each value

can be called by their corresponding index in the list. Here is an example of a list variable definition.

```
variable "users" {
  type = list
  default = ["root", "user1", "user2"]
}
```

Lists can be used in the resource plans similarly to strings, but you'll also need to denote the index of the value

you are looking for.

```
username = var.users[0]
```

Map

Maps are a collection of string keys and string values. These can be useful for selecting values based on

predefined parameters such as the server configuration by the monthly price.

```
variable "plans" {
  type = map
  default = {
    "5USD" = "1xCPU-1GB"
}
```

```
"10USD" = "1xCPU-2GB"
"20USD" = "2xCPU-4GB"
}
}
```

You can access the right value by using the matching key. For example, the variable below would set the plan to "1xCPU-1GB".

```
plan = var.plans["5USD"]
```

The values matching to their keys can also be used to look up information in other maps. For example,

underneath is a shortlist of plans and their corresponding storage sizes.

```
variable "storage_sizes" {
type = map
default = {
"1xCPU-1GB" = "25"
"1xCPU-2GB" = "50"
"2xCPU-4GB" = "80"
}
}
```

These can then be used to find the right storage size based on the monthly price as defined in the previous example.

```
size = lookup(var.storage_sizes, var.plans["5USD"])
```

Boolean

The last of the available variable type is boolean. They give the option to employ simple true or false values.

For example, you might wish to have a variable that decides when to generate the root user password on a new deployment.

```
variable "set_password" {
default = false
}
```

The above example boolean can be used similarly to a string variable by simply marking down the correct variable.

```
create_password = var.set_password
```

By default, the value is set to false in this example. However, you can overwrite the variable at deployment by

assigning a different value in a command-line variable.

```
terraform apply -var set_password="true"
```

## **NEW QUESTION: 71**

Terraform Enterprise currently supports running under which the following operating systems?

- A.** Ubuntu
- B.** Amazon Linux
- C.** Debian
- D.** CentOS
- E.** Red Hat Enterprise Linux
- F.** Oracle Linux

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform Enterprise runs on Linux instances, and you must prepare a running Linux instance for Terraform

Enterprise before running the installer. You will start and manage this instance like any other server.

Terraform Enterprise currently supports running under the following operating systems:

Standalone deployment:

Debian 7.7+

Ubuntu 14.04.5 / 16.04 / 18.04

Red Hat Enterprise Linux 7.4 - 7.8

CentOS 6.x / 7.4 - 7.8

Amazon Linux 2014.03 / 2014.09 / 2015.03 / 2015.09 / 2016.03 / 2016.09 / 2017.03 / 2017.09 / 2018.03 / 2.0

Oracle Linux 7.4 - 7.8

<https://www.terraform.io/docs/enterprise/before-installing/index.html>

## **NEW QUESTION: 72**

John is writing a module and within the module, there are multiple places where he has to use the same

conditional expression but he wants to avoid repeating the same values or expressions multiple times in a

configuration,. What is a better approach to dealing with this?

- A.** Local Values
- B.** Expressions
- C.** Functions
- D.** Variables

**Answer:** ([SHOW ANSWER](#))

Explanation

A local value assigns a name to an expression, allowing it to be used multiple times within a module without repeating it.

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

### **NEW QUESTION: 73**

Your configuration file has been locked accidentally. What of the following command would you use to unlock?

- A. state.tf-unlock
- B. terraform filename-unlock
- C. delete the file and create a new state file
- D. terraform force-unlock

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 74**

If a module declares a variable with a default, that variable must also be defined within the module.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 75**

Which of the following allows Terraform users to apply policy as code to enforce standardized configurations

for resources being deployed via infrastructure as code?

- A. Sentinel
- B. Module registry
- C. Functions
- D. Workspaces

**Answer:** ([SHOW ANSWER](#))

Explanation

Sentinel is a language and framework for policy built to be embedded in existing software to enable

fine-grained, logic-based policy decisions. A policy describes under what circumstances certain behaviors are

allowed. Sentinel is an enterprise-only feature.

[https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb\\_title](https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb_title)

### **NEW QUESTION: 76**

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not

visible in the attributes of a resource. What command will do this?

- A. terraform taint
- B. terraform apply

- C. terraform graph
- D. terraform refresh

**Answer: ([SHOW ANSWER](#))**

Explanation

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as

tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not

visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different

or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For

example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially

new IP address of a tainted server. The plan command will show this if this is the case.

This example will taint a single resource:

```
$ terraform taint aws_security_group.allow_all
```

The resource `aws_security_group.allow_all` in the module root has been marked as tainted.

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

## NEW QUESTION: 77

colleagues is new toTerraform and wants to add a new workspace named new-hire. What command he should

execute from the following?

- A. terraform workspace new new hire
- B. terraform workspace init new-hire

- C. `terraform workspace-new-new-hire`
- D. `terraform workspace new-hire`

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 78**

`terraform refresh` command will not modify infrastructure, but does modify the state file.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

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.

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

### **NEW QUESTION: 79**

Terraform variable names are saved in the state file.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform stores information about your infrastructure in a state file. This state file keeps track of resources

created by your configuration and maps them to real-world resources.

<https://learn.hashicorp.com/tutorials/terraform/state-cli>

### **NEW QUESTION: 80**

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

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

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

### **NEW QUESTION: 81**

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.

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

**Answer:** ([SHOW ANSWER](#))

Explanation

validate will look for syntax errors "Declaring a resource identifier more than once" is a syntax error

### **NEW QUESTION: 82**

A user has created a module called "my\_test\_module" and committed it to GitHub. Over time, several

commits have been made with updates to the module, each tagged in GitHub with an incremental version

number. Which of the following lines would be required in a module configuration block in terraform to select

tagged version v1.0.4?

- A. source = "git::https://example.com/my\_test\_module.git@tag=v1.0.4"
- B. source = "git::https://example.com/my\_test\_module.git&ref=v1.0.4"
- C. source = "git::https://example.com/my\_test\_module.git#tag=v1.0.4"
- D. source = "git::https://example.com/my\_test\_module.git?ref=v1.0.4"

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/modules/sources.html#selecting-a-revision>

### **NEW QUESTION: 83**

Which type of block fetches or computes information for use elsewhere in a Terraform configuration?

- A. provider
- B. resource
- C. local
- D. data

**Answer:** ([SHOW ANSWER](#))

Explanation

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.

## NEW QUESTION: 84

Where does the Terraform local backend store its state?

- A. In the /tmp directory
- B. In the terraform.tfvars file
- C. In the terraform.tfstate file
- D. In the user's .terraformrc file

**Answer:** ([SHOW ANSWER](#))

Explanation

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

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>

## NEW QUESTION: 85

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

**Answer:** ([SHOW ANSWER](#))

Explanation

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  
}
```

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

## NEW QUESTION: 86

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

**Answer:** ([SHOW ANSWER](#))

Explanation

```
resource "aws_instance" "web" {  
    ami= data.aws_ami.ubuntu.id
```

Reference: <https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance>

## NEW QUESTION: 87

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

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/language/syntax/style#style-conventions>

## NEW QUESTION: 88

Named workspaces are not a suitable isolation mechanism for strong separation between staging and

production?

A. True

B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

Organizations commonly want to create a strong separation between multiple deployments of the same

infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In

this case, the backend used for each deployment often belongs to that deployment, with different credentials

and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

## **NEW QUESTION: 89**

You are using a terraform operation that writes state. Unfortunately automatic state unlocking has failed for

that operation. Which of the below commands can be used to remove the already acquired lock on the state?

A. `terraform unlock`

B. `terraform force-unlock`

C. `terraform state unlock`

D. None of the above

**Answer: ([SHOW ANSWER](#))**

Explanation

Command: `force-unlock`

Manually unlock the state for the defined configuration.

This will not modify your infrastructure. This command removes the lock on the state for the current

configuration. The behavior of this lock is dependent on the backend being used. Local state files cannot be

unlocked by another process.

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

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

Terraform has a `force-unlock` command to manually unlock the state if unlocking failed.

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.

## **NEW QUESTION: 90**

John wants to use two different regions to deploy two different EC2 instances. He has specified two provider blocks in his providers.tf file.

```
provider "aws" { region = "us-east-1" }
provider "aws" { region = "us-west-2" }
```

When he run terraform plan he encountered an error. How to fix this?

- A. Use another provider version
- B. Use alias for region = "us-west-2"
- C. It can not be fixed
- D. Use default keyword with region = "us-east-1"

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 91

Complete the following sentence:

The terraform state command can be used to \_\_\_\_\_

- A. modify state
- B. view state
- C. refresh state
- D. There is no such command

**Answer:** ([SHOW ANSWER](#))

Explanation

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 92

In regards to Terraform state file, select all the statements below which are correct?

- A. When using local state, the state file is stored in plain-text.
- B. The state file is always encrypted at rest.
- C. Storing state remotely can provide better security.
- D. Using the mask feature, you can instruct Terraform to mask sensitive data in the state file.
- E. The Terraform state can contain sensitive data, therefore the state file should be protected from unauthorized access.
- F. Terraform Cloud always encrypts state at rest.

**Answer:** ([SHOW ANSWER](#))

## Explanation

Terraform state can contain sensitive data, depending on the resources in use and your definition of

"sensitive." The state contains resource IDs and all resource attributes. For resources such as databases, this

may contain initial passwords.

When using local state, state is stored in plain-text JSON files.

When using remote state, state is only ever held in memory when used by Terraform. It may be encrypted at

rest, but this depends on the specific remote state backend.

Storing Terraform 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.

## Recommendations

If you manage any sensitive data with Terraform (like database passwords, user passwords, or private keys),

treat the state itself as sensitive data.

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.

## NEW QUESTION: 93

Multiple provider instances blocks for AWS can be part of a single configuration file?

A. False

B. True

**Answer: ([SHOW ANSWER](#))**

## Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a

per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud

platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same

provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For

example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it

creates an additional provider configuration. For providers that have no required configuration arguments, the

implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-provider-instances>

## NEW QUESTION: 94

What is the provider for this fictitious resource?

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

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

**Answer: ([SHOW ANSWER](#))**

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

## NEW QUESTION: 95

You have provisioned some aws resources in your test environment through Terraform for a POC work. After

the POC, now you want to destroy the resources but before destroying them you want to check what resources

will be getting destroyed through terraform. what are the options of doing that? (Select TWO)

- A. Use terraform destroy command
- B. This is not possible
- C. Use terraform plan command
- D. Use terraform plan -destroy command.

**Answer: A,D ([LEAVE A REPLY](#))**

Explanation

<https://learn.hashicorp.com/terraform/getting-started/destroy>

### NEW QUESTION: 96

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

**Answer: ([SHOW ANSWER](#))**

Explanation

[https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/device\\_naming.html](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/device_naming.html)

### NEW QUESTION: 97

A junior admin accidentally deleted some of your cloud instances. What does Terraform do when you run

terraform apply?

- A. Rebuild only the instances that were deleted Most Voted
- B. Stop and generate an error message about the missing instances
- C. Tear down the entire workspace infrastructure and rebuild it
- D. Build a completely brand new set of infrastructure

**Answer:** A ([LEAVE A REPLY](#))

### **NEW QUESTION: 98**

True or False. The terraform refresh command is used to reconcile the state Terraform knows about (via its state file) with the real-world infrastructure. If drift is detected between the real-world infrastructure and the last known-state, it will modify the infrastructure to correct the drift.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

Explanation

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

### **NEW QUESTION: 99**

Which of the below command will upgrade the provider version to the latest acceptable one?

- A. terraform plan upgrade
- B. terraform provider -upgrade
- C. terraform init -upgrade
- D. terraform init -update

**Answer:** ([SHOW ANSWER](#))

Explanation

To upgrade to the latest acceptable version of each provider, run terraform init -upgrade. This command also

upgrades to the latest versions of all Terraform modules.

<https://www.terraform.io/docs/configuration/providers.html>

### **NEW QUESTION: 100**

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address aws\_instance.web refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is

used.

C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.

D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer:** ([SHOW ANSWER](#))

Explanation

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:

module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address

applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

aws\_instance.web[3]

Refers to only the last instance in the config, and an address like this:

aws\_instance.web

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

## NEW QUESTION: 101

You have created two workspaces PROD and DEV. You have switched to DEV and provisioned DEV

infrastructure from this workspace. Where is your state file stored?

- A. terraform.d
- B. terraform.tfstate
- C. terraform.tfstate.DEV
- D. terraform.tfstate.d

**Answer:** ([SHOW ANSWER](#))

## Explanation

Terraform stores the workspace states in a directory called `terraform.tfstate.d`. This directory should be treated

similarly to default workspace state file `terraform.tfstate`

`main.tf`

`provider.tf`

`terraform.tfstate.d`

`DEV`

`terraform.tfstate` # DEV workspace state file

`PROD`

`terraform.tfstate` # PROD workspace state file

`terraform.tfvars` # Default workspace state file

`variables.tf`

## NEW QUESTION: 102

True or False: A list(...) contain a number of values of the same type while an object(...) can contain a number of values of different types.

A. False

B. True

**Answer:** ([SHOW ANSWER](#))

## Explanation

### Collection Types

A collection type allows multiple values of one other type to be grouped together as a single value. The type of

value within a collection is called its element type. All collection types must have an element type, which is

provided as the argument to their constructor.

For example, the type `list(string)` means "list of strings", which is a different type than `list(number)`, a list of

numbers. All elements of a collection must always be of the same type.

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.

\* `set(...)`: a collection of unique values that do not have any secondary identifiers or ordering.

<https://www.terraform.io/docs/configuration/types.html>

## Structural Types

A structural type allows multiple values of several distinct types to be grouped together as a single value.

Structural types require a schema as an argument, to specify which types are allowed for which elements.

The two kinds of structural type in the Terraform language are:

\* `object(...)`: a collection of named attributes that each have their own type.

The schema for object types is `{ <KEY> = <TYPE>, <KEY> = <TYPE>, ... }` - a pair of curly braces

containing a comma-separated series of `<KEY> = <TYPE>` pairs. Values that match the object type must

contain all of the specified keys, and the value for each key must match its specified type. (Values with

additional keys can still match an object type, but the extra attributes are discarded during type conversion.)

\* `tuple(...)`: a sequence of elements identified by consecutive whole numbers starting with zero, where each

element has its own type.

The schema for tuple types is `[<TYPE>, <TYPE>, ...]` - a pair of square brackets containing a comma-separated series of types. Values that match the tuple type must have exactly the same number of

elements (no

more and no fewer), and the value in each position must match the specified type for that position.

For example: an object type of `object({ name=string, age=number })` would match a value like the following:

```
{  
  name = "John"  
  age = 52  
}
```

Also, an object type of `object({ id=string, cidr_block=string })` would match the object produced by a

reference to an `aws_vpc` resource, like `aws_vpc.example_vpc`; although the resource has additional attributes,

they would be discarded during type conversion.

Finally, a tuple type of `tuple([string, number, bool])` would match a value like the following:

["a", 15, true]

<https://www.terraform.io/docs/configuration/types.html>

### NEW QUESTION: 103

When using multiple configurations of the same Terraform provider, what meta-argument must be included in any non-default provider configurations?

- A. alias
- B. depends\_on
- C. name
- D. id

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 104

Environment variables can be used to set variables. The environment variables must be in the format

"\_\_\_\_\_"\_<variablename>. Select the correct prefix string from the following list.

- A. TF\_CLI\_ARGS
- B. TF\_VAR
- C. TF\_VAR\_
- D. TF\_VAR\_ENV

**Answer:** ([SHOW ANSWER](#))

Explanation

Environment variables can be used to set variables. The environment variables must be in the format

TF\_VAR\_name and this will be checked last for a value. For example:

```
export TF_VAR_region=us-west-1
export TF_VAR_ami=ami-049d8641
export TF_VAR_alist='[1,2,3]'
export TF_VAR_amap='{ foo = "bar", baz = "qux" }'
```

<https://www.terraform.io/docs/commands/environment-variables.html>

### NEW QUESTION: 105

Multiple configurations for the same provider can be used in a single configuration file.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a

per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud

platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same

provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For

example:

```
# The default provider configuration
```

```
provider "aws" {  
region = "us-east-1"  
}
```

```
# Additional provider configuration for west coast region
```

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

The provider block without alias set is known as the default provider configuration. When alias is set, it

creates an additional provider configuration. For providers that have no required configuration arguments, the

implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-provider-instances>

## NEW QUESTION: 106

When multiple engineers start deploying infrastructure using the same state file, what is a feature of remote

state storage that is critical to ensure the state doesn't become corrupt?

- A. Object Storage
- B. State Locking
- C. WorkSpaces
- D. Encryption

**Answer: ([SHOW ANSWER](#))**

Explanation

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.

State locking happens automatically on all operations that could write state. You won't see any message that it

is happening. If state locking fails, Terraform will not continue. You can disable state locking for most

commands with the `-lock` flag but it is not recommended.

If acquiring the lock is taking longer than expected, Terraform will output a status message. If Terraform

doesn't output a message, state locking is still occurring if your backend supports it.

Not all backends support locking. Please view the list of backend types for details on whether a backend

supports locking or not.

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 107

Which of the following is not valid source path for specifying a module?

- A. `source = "./moduleversion=v1.0.0"`
- B. `source = "hashicorp/consul/aws"`
- C. `source = "./module"`
- D. `source = "github.com/hashicorp/example?ref=v1.0.0"`

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 108

A terraform apply can not \_\_\_\_\_ infrastructure.

- A. `destroy`
- B. `import`
- C. `provision`
- D. `change`

Answer: B ([LEAVE A REPLY](#))

### NEW QUESTION: 109

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

**Answer: ([SHOW ANSWER](#))**

Explanation

"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 actions need to be taken."

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

**NEW QUESTION: 110**

In order to reduce the time it takes to provision resources, Terraform uses parallelism. By default, how many

resources will Terraform provision concurrently?

- A. 10
- B. 20
- C. 50
- D. 5

**Answer: ([SHOW ANSWER](#))**

**NEW QUESTION: 111**

Please identify the offerings which are unique to Terraform Enterprise, and not available in either Terraform

OSS, or Terraform Cloud. Select four.

- A. Audit Logs
- B. Private Network Connectivity
- C. VCS Integration
- D. Sentinel
- E. Clustering

**Answer: ([SHOW ANSWER](#))**

## Explanation

<https://www.hashicorp.com/products/terraform/pricing/>

### NEW QUESTION: 112

From the code below, identify the implicit dependency:

- A. The EC2 instance labeled web\_server
- B. The S3 bucket labeled company\_data
- C. The EIP with an id of ami-2757f631
- D. The AMI used for the EC2 instance

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 113

Terraform is currently being used by your organisation to create resources on AWS for the development of a

web application. One of your coworkers wants to change the instance type to "t2.large" while keeping the default set values.

What adjustments does the teammate make in order to meet his goal?

- A. Modify the tf.variables with the instance type and issue terraform apply
- B. Issue Terraform plan instance.type".t2.large" and it deploys the instance
- C. Modify the terraform.tfvars with the instance type and issue terraform plan and then terraform apply to deploy the instances
- D. Create a new file my.tfvars and add the type of the instance and issue terraform plan and apply

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 114

While using generic git repository as a module source, which of the below options allows terraform to select a specific version or tag instead of selecting the HEAD.

- A. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?ref=v1.2.0"}
```

- B. Append version argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?version=v1.2.0"}
```

- C. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git#ref=v1.2.0"}
```

- D. By default, Terraform will clone and use the default branch (referenced by HEAD) in the selected repository and you can not override this.

Answer: ([SHOW ANSWER](#))

## Explanation

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:

```
module "vpc" {  
  source = "git::https://example.com/vpc.git?ref=v1.2.0"  
}
```

The value of the ref argument can be any reference that would be accepted by the git checkout command,

including branch and tag names.

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

## NEW QUESTION: 115

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.

**Answer: A,D ([LEAVE A REPLY](#))**

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

## NEW QUESTION: 116

What are some of the features of Terraform state? (select three)

- A. inspection of cloud resources
- B. determining the correct order to destroy resources
- C. mapping configuration to real-world resources
- D. increased performance

**Answer: ([SHOW ANSWER](#))**

## NEW QUESTION: 117

True or False: Workspaces provide identical functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.

- A. True

**B.** False

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/cloud/workspaces/index.html>

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

### NEW QUESTION: 118

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

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/language/meta-arguments/count#referring-to-instances>

Reference: <https://www.terraform.io/docs/configuration-0-11/interpolation.html>

### NEW QUESTION: 119

You want to know from which paths Terraform is loading providers referenced in your Terraform configuration (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

**Answer:** ([SHOW ANSWER](#))

Explanation

Although this will only output to stderr and if you need to review log file you will need to include TF\_LOG\_PATH=pathToFile <https://www.terraform.io/internals/debugging>

### NEW QUESTION: 120

If you enable TF\_LOG = DEBUG, the log will be stored in syslog.log file in the correct directory.

- A.** False

- B.** True

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/internals/debugging.html>

### NEW QUESTION: 121

What Terraform command can be used to inspect the current state file?

- A. terraform read
- B. terraform inspect
- C. terraform show
- D. terraform state

Answer: ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 122

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 not make any changes to the Load Balancer and will update the state file to reflect any changes made.
- B. Terraform will change the load balancer port to 80, and) then change it back to 443
- C. Terraform will fail with an error because the state file is no longer accurate
- D. Terraform will change the port back to 80 in your code

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 123

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

**Answer:** ([SHOW ANSWER](#))

Explanation

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

## NEW QUESTION: 124

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"

**Answer:** ([SHOW ANSWER](#))

Explanation

version = ">= 1.2.0, < 2.0.0"

A version constraint is a string literal containing one or more conditions, which are separated by commas.

Each condition consists of an operator and a version number.

Version numbers should be a series of numbers separated by periods (like 1.2.0), optionally with a suffix to

indicate a beta release.

The following operators are valid:

= (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 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://www.terraform.io/language/expressions/version-constraints>

### NEW QUESTION: 125

A user creates three workspaces from the command line - prod, dev, and test. Which of the following

commands will the user run to switch to the dev workspace?

- A. `terraform workspace dev`
- B. `terraform workspace select dev`
- C. `terraform workspace -switch dev`
- D. `terraform workspace switch dev`

**Answer:** B ([LEAVE A REPLY](#))

Explanation

The `terraform workspace select` command is used to choose a different workspace to use for further operations.

<https://www.terraform.io/docs/commands/workspace/select.html>

### NEW QUESTION: 126

Which of the following is allowed as a Terraform variable name?

- A. count
- B. name
- C. source
- D. version

**Answer:** ([SHOW ANSWER](#))

Explanation

"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>

### NEW QUESTION: 127

The following is a snippet from a Terraform configuration file:

Which, when validated, results in the following error:

Fill in the blank in the error message with the correct string from the list below.

- A. `version`
- B. `multi`
- C. `label`
- D. `alias`

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-providerinstances>

### **NEW QUESTION: 128**

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

**Answer:** ([SHOW ANSWER](#))

Explanation

"The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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>

### **NEW QUESTION: 129**

You have just developed a new Terraform configuration for two virtual machines with a cloud provider. You

would like to create the infrastructure for the first time.

Which Terraform command should you run first?

- A. terraform init
- B. terraform plan
- C. terraform show
- D. terraform apply

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 130**

Which command lets you experiment with Terraform's built-in functions?

- A. terraform env
- B. terraform console
- C. terraform test
- D. terraform validate

**Answer:** ([SHOW ANSWER](#))

Explanation

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

### **NEW QUESTION: 131**

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

**Answer:** ([SHOW ANSWER](#))

Explanation

The terraform fmt command is used to rewrite Terraform configuration files to a canonical format and style.

### **NEW QUESTION: 132**

Consider the following Terraform 0.12 configuration snippet:

```
1. variable "vpc_cidrs" {  
2.   type = map  
3.   default = {  
4.     us-east-1 = "10.0.0.0/16"  
5.     us-east-2 = "10.1.0.0/16"  
6.     us-west-1 = "10.2.0.0/16"  
7.     us-west-2 = "10.3.0.0/16"  
8.   }  
9. }  
10.  
11. resource "aws_vpc" "shared" {  
12.   cidr_block = _____  
13. }
```

How would you define the cidr\_block for us-east-1 in the aws\_vpc resource using a variable?

- A. var.vpc\_cidrs["us-east-1"]
- B. var.vpc\_cidrs.0
- C. vpc\_cidrs["us-east-1"]
- D. var.vpc\_cidrs[0]

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 133**

What Terraform feature is shown in the example below?

- A. conditional expression
- B. local values
- C. dynamic block
- D. data source

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 134**

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A. Explicit use of a provider block in configuration, optionally including a version constraint.
- B. Use of any resource belonging to a particular provider in a resource or data block in configuration.
- C. Existence of any resource instance belonging to a particular provider in the current state.
- D. Existence of any provider plugins found locally in the working directory.

**Answer:** ([SHOW ANSWER](#))

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider

dependency. The plugin can exist without any reference to it in the terraform configuration.

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

### **NEW QUESTION: 135**

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

**Answer:** ([SHOW ANSWER](#))

Explanation

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

### **NEW QUESTION: 136**

What is the name assigned by Terraform to reference this resource?

```
resource "azurerm_resource_group" "dev" {  
    name = "test"  
    location = "westus"  
}
```

- A. test
- B. azurerm\_resource\_group
- C. dev
- D. azurerm

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

### NEW QUESTION: 137

What is a downside to using the Vault provider to read secrets from Vault?

- A. Secrets are persisted to the state file and plans.
- B. Terraform and Vault must be running on the same version.
- C. Terraform and Vault must be running on the same physical host.
- D. Terraform requires a unique auth method to work with Vault.

**Answer:** ([SHOW ANSWER](#))

Explanation

The Vault provider allows Terraform to read from, write to, and configure Hashicorp Vault.

Interacting with Vault from Terraform causes any secrets that you read and write to be persisted in both

Terraform's state file and in any generated plan files. For any Terraform module that reads or writes Vault

secrets, these files should be treated as sensitive and protected accordingly.

### NEW QUESTION: 138

Your team has started using terraform OSS in a big way , and now wants to deploy multi region deployments

(DR) in aws using the same terraform files . You want to deploy the same infra (VPC,EC2 ...) in both

us-east-1 ,and us-west-2 using the same script , and then peer the VPCs across both the regions to enable DR

traffic. But , when you run your script , all resources are getting created in only the default provider region.

What should you do? Your provider setting is as below -

```
# The default provider configuration provider "aws" { region = "us-east-1" }
```

A. No way to enable this via a single script . Write 2 different scripts with different default providers in the

2 scripts , one for us-east , another for us-west.

B. Create a list of regions , and then use a for-each to iterate over the regions , and create the same

resources ,one after the one , over the loop.

C. Use provider alias functionality , and add another provider for us-west region . While creating the

resources using the tf script , reference the appropriate provider (using the alias).

**D.** Manually create the DR region , once the Primary has been created , since you are using terraform OSS ,

and multi region deployment is only available in Terraform Enterprise.

**Answer:** ([SHOW ANSWER](#))

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a

per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud

platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same

provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For

example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

<https://www.terraform.io/docs/configuration/providers.html>

## NEW QUESTION: 139

You wanted to destroy some of the dependent resources from real infrastructure. You choose to delete those

resources from your configuration file and run terraform plan and then apply. Which of the following way

your resources would be destroyed?

**A.** Terraform can still determine the correct order for destruction from the state even when you delete one

or more items from the configuration.

**B.** Those would be destroyed in the order in which they were written in the configuration file previously

before you have deleted them from configuration file.

**C.** The resource will be destructed in random order as you have already deleted them from configuration.

- D. You can not destroy resources by deleting them from configuration file and running plan and apply.

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform typically uses the configuration to determine dependency order. However, when you delete a

resource from a Terraform configuration, Terraform must know how to delete that resource.

Terraform can see

that a mapping exists for a resource not in your configuration and plan to destroy. However, since the

configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or

more items from the configuration.

## NEW QUESTION: 140

What does terraform destroy do?

~~remove everything~~

- 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

**Answer:** ([SHOW ANSWER](#))

Explanation

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>

## NEW QUESTION: 141

What allows you to conveniently switch between multiple instances of a single configuration within its single backend?

- A. Local backends
- B. Providers
- C. Remote backends
- D. Workspaces

**Answer:** ([SHOW ANSWER](#))

## Explanation

Named workspaces allow conveniently switching between multiple instances of a single configuration within its single backend. ... A common use for multiple workspaces is to create a parallel, distinct copy of a set of infrastructure in order to test a set of changes before modifying the main production infrastructure. Workspaces, allowing multiple states to be associated with a single configuration. The configuration still has only one backend, but multiple distinct instances of that configuration to be deployed without configuring a new backend or changing authentication credentials.

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

## NEW QUESTION: 142

What advantage does an operations team that uses infrastructure as code have?

- A. The ability to delete infrastructure
- B. The ability to autoscale a group of servers
- C. The ability to update existing infrastructure
- D. The ability to reuse best practice configurations and settings

Answer: ([SHOW ANSWER](#))

## NEW QUESTION: 143

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

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

Answer: E ([LEAVE A REPLY](#))

## Explanation

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.

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

## NEW QUESTION: 144

While Terraform is generally written using the HashiCorp Configuration Language (HCL), what other syntax can Terraform be expressed in?

- A. JSON
- B. YAML
- C. TypeScript
- D. XML

**Answer:** ([SHOW ANSWER](#))

Explanation

The constructs in the Terraform language can also be expressed in JSON syntax, which is harder for humans to read and edit but easier to generate and parse programmatically.

### NEW QUESTION: 145

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.

**Answer:**

depends\_on

### NEW QUESTION: 146

Workspaces in Terraform provides similar functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/cloud/migrate/workspaces.html>

Workspaces, managed with the `terraform workspace` command, aren't the same thing as Terraform Cloud's

workspaces. Terraform Cloud workspaces act more like completely separate working directories; CLI workspaces are just alternate state files.

### NEW QUESTION: 147

Dawn has created the below child module. Without changing the module, can she override the `instance_type`

from t2.micro to t2.large from her code while calling this module?

```
1. resource "aws_instance" "myec2"
2. {
3.   ami = "ami-082b5a644766e0e6f"
4.   instance_type = "t2.micro"
5. }
```

A. YES

B. No

**Answer:** ([SHOW ANSWER](#))

Explanation

As the instance\_type is hard-coded in source module, you will not be able to change its value from destination

module. Instead of hard-coding you should use variable with default values.

### **NEW QUESTION: 148**

You have created a main.tr 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 prompt you to pick which resource you want to destroy
- B. Terraform will immediately destroy all the infrastructure
- C. Terraform will prompt you to confirm that you want to destroy all the infrastructure
- D. Terraform will destroy the application server because it is listed first in the code
- E. Terraform will destroy the main.tf file

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 149**

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
- D. By default, it does not refresh your state file to reflect current infrastructure configuration
- E. You cannot target specific resources for the operation

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/cli/run>

### **NEW QUESTION: 150**

Your company has a lot of workloads in AWS , and Azure that were respectively created using CloudFormation , and AzureRM Templates. However , now your CIO has decided to use Terraform for all new projects , and has asked you to check how to integrate the existing environment with terraform code.

What should be your next plan of action?

- A. This is only possible in Terraform Enterprise , which has the TerraformConverter exe that can take any other template language like AzureRM and convert to Terraform code.
- B. Use terraform import command to import each resource one by one .
- C. Tell the CIO that this is not possible . Resources created in CloudFormation , and AzureRM templates cannot be tracked using terraform.
- D. Just write the terraform config file for the new resources , and run terraform apply , the state file will automatically be updated with the details of the new resources to be imported.

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 151**

Which of the following best describes a Terraform provider?

- A. A plugin that Terraform uses to translate the API interactions with the service or provider.
- B. Serves as a parameter for a Terraform module that allows a module to be customized.
- C. Describes an infrastructure object, such as a virtual network, compute instance, or other components.
- D. A container for multiple resources that are used together.

**Answer:** ([SHOW ANSWER](#))

Explanation

A provider is responsible for understanding API interactions and exposing resources. Providers generally are

an IaaS (e.g. Alibaba Cloud, AWS, GCP, Microsoft Azure, OpenStack), PaaS (e.g. Heroku), or SaaS services  
(e.g. Terraform Cloud, DNSimple, Cloudflare).

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the newest Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (450 Q&As Dumps, **30%OFF Special Discount: freecram**)

## **NEW QUESTION: 152**

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

**Answer: ([SHOW ANSWER](#))**

Explanation

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

## **NEW QUESTION: 153**

Select two answers to complete the following sentence: Before a new provider can be used, it must be \_\_\_\_\_

and \_\_\_\_\_.

- A. approved by HashiCorp
- B. uploaded to source control
- C. declared in the configuration
- D. initialized

**Answer: ([SHOW ANSWER](#))**

Explanation

Each time a new provider is added to configuration -- either explicitly via a provider block or by adding a

resource from that provider -- Terraform must initialize the provider before it can be used.

Initialization

downloads and installs the provider's plugin so that it can later be executed.

## **NEW QUESTION: 154**

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A. TF\_TRACE
- B. TF\_DEBUG
- C. TF\_LOG
- D. TF\_INFO

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

### **NEW QUESTION: 155**

When Terraform needs to be installed in a location where it does not have internet access to download the

installer and upgrades, the installation is generally known as to be \_\_\_\_\_.

- A. air-gapped
- B. disconnected
- C. a private install
- D. non-traditional

**Answer: ([SHOW ANSWER](#))**

Explanation

A Terraform Enterprise install that is provisioned on a network that does not have Internet access is generally

known as an air-gapped install. These types of installs require you to pull updates, providers, etc. from external

sources vs. being able to download them directly.

### **NEW QUESTION: 156**

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

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

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-prod). 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.

### NEW QUESTION: 157

You have already set TF\_LOG = DEBUG to enable debug log. Now you want to always write the log to the

directory you're currently running terraform from. what should you do to achieve this.

- A. Run the command export TF\_LOG\_FILE=./terraform.log.
- B. Run the command export TF\_LOG\_PATH=./terraform.log.
- C. Run the command export TF\_DEBUG\_PATH=./terraform.log.
- D. No explicit action required. Terraform will take care of this as you have enable TF\_LOG.

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/commands/environment-variables.html>

### NEW QUESTION: 158

Which one of the following will run echo 0 and echo 1 on a newly created host?

- A. provisioner "local-exec" { command = "echo 0"  
command = "echo 1"  
}
- B. provisioner "remote-exec" {  
inline = [  
echo 0,  
echo 1  
]  
}
- C. provisioner "remote-exec" {  
command = "\${echo 0}"  
command = "\${echo 1}"  
}
- D. provisioner "remote-exec" {  
inline = [  
"echo 0",  
"echo 1"  
]  
}

**Answer:** ([SHOW ANSWER](#))

Explanation

remote-exec Provisioner

Example usage

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

```
provisioner "remote-exec" {
  inline = [
    "puppet apply",
    "consul join ${aws_instance.web.private_ip}",
  ]
}
```

### NEW QUESTION: 159

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. Make the change programmatically via the public cloud CLI
- B. Make the change via the public cloud API endpoint
- C. Use the public cloud console to make the change after a database record has been approved
- D. Submit a pull request and wait for an approved merge of the proposed changes
- E. Clone the repository containing your infrastructure code and then run the code

Answer: ([SHOW ANSWER](#))

### NEW QUESTION: 160

From the answers below, select the advantages of using Infrastructure as Code.

- A. Provide a codified workflow to develop customer-facing applications.
- B. Safely test modifications using a "dry run" before applying any actual changes.
- C. Easily integrate with application workflows (GitLab Actions, Azure DevOps, CI/CD tools).
- D. Easily change and update existing infrastructure.
- E. Provide reusable modules for easy sharing and collaboration.

Answer: ([SHOW ANSWER](#))

Explanation

Infrastructure as Code is not used to develop applications, but it can be used to help deploy or provision those

applications to a public cloud provider or on-premises infrastructure.

All of the others are benefits to using Infrastructure as Code over the traditional way of managing infrastructure, regardless if it's public cloud or on-premises.

### NEW QUESTION: 161

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

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/cli/commands/refresh#:~:text=The%20terraform%20refresh%20command%20reads%2>

## NEW QUESTION: 162

Which of the following actions are performed during a terraform init?

- A. Initializes downloaded and/or installed providers
- B. Initializes the backend configuration
- C. Provisions the declared resources in your configuration
- D. Download the declared providers which are supported by HashiCorp

**Answer:** ([SHOW ANSWER](#))

Explanation

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

This is the first command that should be run after writing a new Terraform configuration or cloning an existing

one from version control. It is safe to run this command multiple times.

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.

terrafrom init command does -

- \* Copy a Source Module
- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

## NEW QUESTION: 163

Which of the below options is a valid interpolation syntax for retrieving a data source?

- A. \${google\_storage\_bucket.backend}
- B. \${azurerm\_resource\_group.test.data}
- C. \${aws\_instance.web.id.data}
- D. \${data.google\_dns\_keys.foo\_dns\_keys.key\_signing\_keys[0].ds\_record}

**Answer:** ([SHOW ANSWER](#))

Explanation

Data source attributes are interpolated with the general syntax `data.TYPE.NAME.ATTRIBUTE`.

The

interpolation for a resource is the same but without the `data.` prefix (`TYPE.NAME.ATTRIBUTE`).

<https://www.terraform.io/docs/configuration-0-11/interpolation.html#attributes-of-a-data-source>

### NEW QUESTION: 164

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

**Answer:** ([SHOW ANSWER](#))

Explanation

`-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>

### NEW QUESTION: 165

By default, a defined provisioner is a creation-time provisioner.

- A. True
- B. False

**Answer: A** ([LEAVE A REPLY](#))

Explanation

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated and answers have been corrected** get the newest Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (**450 Q&As Dumps, 30%OFF Special Discount: freecram**)

# HashiCorp.TA-002-P.v2022-05-25.q177

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	177
<b>Version:</b>	v2022-05-25
<b># of views:</b>	120
<b># of Questions views:</b>	1849

<https://www.freecram.net/torrent/HashiCorp.TA-002-P.v2022-05-25.q177.html>

## NEW QUESTION: 1

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

**Answer:** ([SHOW ANSWER](#))

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.

## NEW QUESTION: 2

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

**Answer:** ([SHOW ANSWER](#))

Backends are configured with a nested backend block within the top-level terraform block.

## NEW QUESTION: 3

Which of the following statements best describes the Terraform list(...) type?

- A. a sequence of values identified by consecutive whole numbers starting with zero.
- B. a collection of unique values that do not have any secondary identifiers or ordering.
- C. a collection of named attributes that each have their own type.

Explanation

A terraform list is a sequence of values identified by consecutive whole numbers starting with zero. <https://www.terraform.io/docs/configuration/types.html#structural-types>

- D. a collection of values where each is identified by a string label.

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 4**

Your team uses terraform OSS . You have created a number of reusable modules for important , independent network components that you want to share with your team to enhance consistency . What is the correct option/way to do that?

- A. Terraform modules cannot be shared in OSS version . Each developer needs to maintain their own modules and leverage them in the main tf file.
- B. Upload your modules with proper versioning in the terraform public module registry . Terraform OSS is directly integrated with the public module registry , and can reference the modules from the code in the main tf file.
- C. Terraform module sharing is only available in Enterprise version via terraform private module registry , so no way to enable it in OSS version.
- D. Store your modules in a NAS/ shared file server , and ask your team members to directly reference the code from there. This is the only viable option in terraform OSS ,which is better than individually maintaining module versions for every developer.

**Answer:** ([SHOW ANSWER](#))

Explanation

Software development encourages code reuse through reusable artifacts, such as libraries, packages and modules. Most programming languages enable developers to package and publish these reusable components and make them available on a registry or feed. For example, Python has Python Package Index and PowerShell has PowerShell Gallery.

For Terraform users, the Terraform Registry enables the distribution of Terraform modules, which are reusable configurations. The Terraform Registry acts as a centralized repository for module sharing, making modules easier to discover and reuse.

The Registry is available in two variants:

- \* Public Registry houses official Terraform providers -- which are services that interact with an API to expose and manage a specific resource -- and community-contributed modules.
- \* Private Registry is available as part of the Terraform Cloud, and can host modules internally within an organization.

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

#### **NEW QUESTION: 5**

Complete the following sentence:

The terraform state command can be used to \_\_\_\_\_

- A. modify state
- B. view state
- C. refresh state
- D. There is no such command

**Answer: ([SHOW ANSWER](#))**

Explanation

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

### **NEW QUESTION: 6**

You have declared an input variable called environment in your parent module. What must you do to pass the value to a child module in the configuration?

- A. Nothing, child modules inherit variables of parent module
- B. Add node\_count = var.node\_count
- C. Declare a node\_count input variable for child module
- D. Declare the variable in a terraform.tfvars file

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 7**

Which of the following actions are performed during a terraform init?

- A. Initializes downloaded and/or installed providers
- B. Initializes the backend configuration
- C. Provisions the declared resources in your configuration
- D. Download the declared providers which are supported by HashiCorp

**Answer: ([SHOW ANSWER](#))**

The terraform init command is used to initialize a working directory containing Terraform configuration files. This is the first command that should be run after writing a new

Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times.

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.

terraform init command does -

- \* Copy a Source Module
- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

### **NEW QUESTION: 8**

You have written a terraform IaC script which was working till yesterday , but is giving some vague error from today , which you are unable to understand . You want more detailed logs that could potentially help you troubleshoot the issue , and understand the root cause. What can you do to enable this setting? Please note , you are using terraform OSS.

- A. Terraform OSS can push all its logs to a syslog endpoint. As such, you have to set up the syslog sink, and enable TF\_LOG\_PATH env variable to the syslog endpoint and all logs will automatically start streaming.
- B. Detailed logs are not available in terraform OSS, except the crash message. You need to upgrade to terraform enterprise for this point.
- C. Enable the TF\_LOG\_PATH to the log sink file location, and logging output will automatically be stored there.
- D. Enable TF\_LOG to the log level DEBUG, and then set TF\_LOG\_PATH to the log sink file location.

Terraform debug logs will be dumped to the sink path, even in terraform OSS.

**Answer:** ([SHOW ANSWER](#))

Explanation

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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

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.

## NEW QUESTION: 9

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

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

**Answer:** ([SHOW ANSWER](#))

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>

## NEW QUESTION: 10

- A. terraform workspace
- B. terraform workspace show

- C. `terraform workspace list`
- D. `terraform show workspace`

**Answer: ([SHOW ANSWER](#))**

`terraform workspace list`

The command will list all existing workspaces.

### **NEW QUESTION: 11**

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. Terraform attempts to provision the resource up to three times before exiting with an error
- B. the terraform plan is rolled back and all provisioned resources are removed
- C. it is automatically deleted
- D. the resource is marked as tainted

**Answer: ([SHOW ANSWER](#))**

If a resource successfully creates but fails during provisioning, Terraform will error and mark the resource as "tainted". A resource that is tainted has been physically created, but can't be considered safe to use since provisioning failed. Terraform also does not automatically roll back and destroy the resource during the apply when the failure happens, because that would go against the execution plan: the execution plan would've said a resource will be created, but does not say it will ever be deleted.

### **NEW QUESTION: 12**

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

- A. `replace`
- B. `format`
- C. `join`
- D. `tostring`

**Answer: ([SHOW ANSWER](#))**

Explanation

<https://www.terraform.io/docs/configuration/functions/tostring.html>

### **NEW QUESTION: 13**

What does the default "local" Terraform backend store?

- A. `tfplan` files
- B. Terraform binary
- C. Provider plugins
- D. State file

**Answer: ([SHOW ANSWER](#))**

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

## **NEW QUESTION: 14**

How does Terraform handle working with so many providers?

- A. Terraform ships with all of the plugins embedded in the Terraform binary.
- B. Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in the configuration's working directory.
- C. Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in a shared, system-wide plugins directory.
- D. Terraform allows you to select the providers you want to support during the Terraform installation process.

**Answer:** ([SHOW ANSWER](#))

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

## **NEW QUESTION: 15**

Which of the following is not an action performed by terraform init?

- A. Initialize a configured backend
- B. Retrieve the source code for all referenced modules
- C. Create a sample main.tf file
- D. Load required provider plugins

**Answer:** ([SHOW ANSWER](#))

## **NEW QUESTION: 16**

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?

- A. aws\_ami.ubuntu
- B. data.aws\_ami.ubuntu
- C. data.aws\_ami.ubuntu.id
- D. aws\_ami.ubuntu.id

**Answer:** ([SHOW ANSWER](#))

```
resource "aws_instance" "web" {  
  ami = data.aws_ami.ubuntu.id
```

Reference:

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

### NEW QUESTION: 17

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A. Explicit use of a provider block in configuration, optionally including a version constraint.
- B. Use of any resource belonging to a particular provider in a resource or data block in configuration.
- C. Existence of any resource instance belonging to a particular provider in the current state.
- D. Existence of any provider plugins found locally in the working directory.

**Answer:** ([SHOW ANSWER](#))

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider dependency. The plugin can exist without any reference to it in the terraform configuration. <https://www.terraform.io/docs/commands/providers.html>

### NEW QUESTION: 18

ABC Enterprise has recently tied up with multiple small organizations for exchanging database information.

Due to this, the firewall rules are increasing and are more than 100 rules. This is leading firewall configuration file that is difficult to manage. What is the way this type of configuration can be managed easily?

- A. Terraform Backends
- B. Terraform Expression
- C. Terraform Functions
- D. Dynamic Blocks

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 19

What kind of resource dependency is stored in terraform.tfstate file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

## **Answer: ([SHOW ANSWER](#))**

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

## **NEW QUESTION: 20**

What does this code do?

- A. Requires any version of the AWS provider > = 3.0 and < 4.0
- B. Requires any version of the AWS provider after the 3.0 major release like 4.1
- C. Requires any version of the AWS provider > 3.0
- D. Requires any version of the AWS provider > = 3.0

## **Answer: ([SHOW ANSWER](#))**

## **NEW QUESTION: 21**

You cannot publish your own modules on the Terraform Registry.

- A. False
- B. True

## **Answer: ([SHOW ANSWER](#))**

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

## **NEW QUESTION: 22**

Valarie has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code. Valarie wants to hide the output value in the CLI after terraform apply that's why she has used sensitive parameter.

1. output "db\_password" {
2. value = local.db\_password
3. sensitive = true
4. }

Since sensitive is set to true, will the value associated with db password be available in plain-text in the state file for everyone to read?

- A. Yes
- B. No

## **Answer: ([SHOW ANSWER](#))**

Outputs can be marked as containing sensitive material by setting the sensitive attribute to true, like this:

```
output "sensitive" {  
    sensitive = true  
    value = VALUE
```

}

When outputs are displayed on-screen following a terraform apply or terraform refresh, sensitive outputs are redacted, with <sensitive> displayed in place of their value.

#### Limitations of Sensitive Outputs

The values of sensitive outputs are still stored in the Terraform state, and available using the terraform output command, so cannot be relied on as a sole means of protecting values.

Sensitivity is not tracked internally, so if the output is interpolated in another module into a resource, the value will be displayed.

#### NEW QUESTION: 23

Which of the following statements about local modules is incorrect:

- A. None of the above (all statements above are correct)
- B. Local modules are sourced from a directory on disk
- C. Local modules support versions
- D. Local modules are not cached by terraform init command
- E. All of the above (all statements above are incorrect)

**Answer: C ([LEAVE A REPLY](#))**

#### NEW QUESTION: 24

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. Inspect all Terraform outputs to make sure they are correct
- B. Run an apply and confirm the planned changes
- C. Reinitialize because your configuration has changed
- D. Inspect your Terraform state because you want to change it

**Answer: ([SHOW ANSWER](#))**

#### NEW QUESTION: 25

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

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

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

#### NEW QUESTION: 26

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

**Answer:** ([SHOW ANSWER](#))

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>

## NEW QUESTION: 27

The current implementation of Terraform import can only import resources into the state. It does not generate configuration.

A. False

B. True

**Answer:** ([SHOW ANSWER](#))

Explanation

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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/docs/import/index.html#currently-state-only>

## NEW QUESTION: 28

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

**Answer:** B ([LEAVE A REPLY](#))

The variable declaration can also include a default argument.

## NEW QUESTION: 29

John is writing a module and within the module, there are multiple places where he has to use the same conditional expression but he wants to avoid repeating the same values or expressions multiple times in a configuration,. What is a better approach to dealing with this?

- A. Local Values
- B. Expressions
- C. Functions
- D. Variables

**Answer:** ([SHOW ANSWER](#))

Explanation

A local value assigns a name to an expression, allowing it to be used multiple times within a module without repeating it.

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

### NEW QUESTION: 30

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

**Answer:** ([SHOW ANSWER](#))

Reference: <https://www.terraform.io/docs/configuration-0-11/interpolation.html>

### NEW QUESTION: 31

Anyone can publish and share modules on the Terraform Public Module Registry, and meeting the requirements for publishing a module is extremely easy. Select from the following list all valid requirements. (select three)

- A. The module must be PCI/HIPPA compliant.
- B. Module repositories must use this three-part name format, terraform-- .
- C. The registry uses tags to identify module versions.
- D. Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .
- E. The module must be on GitHub and must be a public repo.

**Answer:** ([SHOW ANSWER](#))

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

### NEW QUESTION: 32

Setting the TF\_LOG environment variable to DEBUG causes debug messages to be logged into syslog.

A. True

B. False

**Answer:** ([SHOW ANSWER](#))

Explanation/Reference: <https://www.terraform.io/docs/internals/debugging.html>

### NEW QUESTION: 33

The Security Operations team of ABC Enterprise wants to mandate that all the Terraform configuration that creates an S3 bucket must have encryption feature enabled. What is the best way to achieve it?

A. Use Sentinel Policies.

B. Use S3 bucket policy.

C. Create a script that checks the encryption parameter is enabled on every git commit.

D. Shared a SOP to engineers to mandate encryption feature on S3.

**Answer:** ([SHOW ANSWER](#))

Explanation

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

Using Sentinel with Terraform Cloud involves:

\* Defining the policies - Policies are defined using the policy language with imports for parsing the Terraform plan, state and configuration.

\* Managing policies for organizations - Users with permission to manage policies can add policies to their organization by configuring VCS integration or uploading policy sets through the API. They also define which workspaces the policy sets are checked against during runs. (More about permissions.)

\* 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.

\* Mocking Sentinel Terraform data - Terraform Cloud provides the ability to generate mock data for any run within a workspace. This data can be used with the Sentinel CLI to test policies before deployment.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### **NEW QUESTION: 34**

terraform refresh command will not modify infrastructure, but does modify the state file.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

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.

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

### **NEW QUESTION: 35**

The current implementation of Terraform import can only import resources into the state. It does not generate configuration.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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/docs/import/index.html#currently-state-only>

### **NEW QUESTION: 36**

Terraform must track metadata such as resource dependencies. Where is this data stored?

- A. workspace
- B. backend
- C. state file
- D. metadata store

**Answer:** ([SHOW ANSWER](#))

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state. Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.  
<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 37**

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

You can change your backend configuration at any time. You can change both the configuration itself as well as the type of backend (for example from "consul" to "s3"). Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

### **NEW QUESTION: 38**

Which of the following is an invalid variable name?

- A. count
- B. web
- C. var1
- D. instance\_name

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/intro/examples/count.html>

### **NEW QUESTION: 39**

True or False? Each Terraform workspace uses its own state file to manage the infrastructure associated with that particular workspace.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

Explanation

The persistent data stored in the backend belongs to a workspace. Initially, the backend has only one workspace, called "default", and thus there is only one Terraform state associated with that configuration.

#### **NEW QUESTION: 40**

- A. To invoke a local executable.
- B. Executes a command on the resource to invoke an update to the Terraform state.
- C. To execute one or more commands on the machine running Terraform.
- D. Ensures that the resource is only executed in the local infrastructure where Terraform is deployed.

**Answer:** ([SHOW ANSWER](#))

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.

Note that even though the resource will be fully created when the provisioner is run, there is no guarantee that it will be in an operable state - for example system services such as sshd may not be started yet on compute resources.

Example usage

```
resource "aws_instance" "web" {  
# ...  
provisioner "local-exec" {  
command = "echo ${aws_instance.web.private_ip} >> private_ips.txt"  
}  
}
```

Note: Provisioners should only be used as a last resort. For most common situations there are better alternatives.

<https://www.terraform.io/docs/provisioners/local-exec.html>

#### **NEW QUESTION: 41**

Which of the following connection types are supported by the remote-exec provisioner?  
(select two)

- A. WinRM
- B. UDP
- C. SMB
- D. RDP
- E. ssh

**Answer:** ([SHOW ANSWER](#))

Explanation

The remote-exec provisioner invokes a script on a remote resource after it is created. The remote-exec provisioner supports both ssh and winrm type connections.

remote-exec connection types -

\* ssh on Linux

\* winrm on Windows

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

## NEW QUESTION: 42

You have declared a variable name my\_var in terraform configuration without a value associated with it.

variable my\_var {}

After running terraform plan it will show an error as variable is not defined.

A. True

B. False

**Answer: ([SHOW ANSWER](#))**

Input variables are usually defined by stating a name, type and a default value. However, the type and default values are not strictly necessary. Terraform can deduct the type of the variable from the default or input value.

Variables can be predetermined in a file or included in the command-line options. As such, the simplest variable is just a name while the type and value are selected based on the input.

variable "variable\_name" {}

terraform apply -var variable\_name="value"

The input variables, like the one above, use a couple of different types: strings, lists, maps, and boolean. Here are some examples of how each type are defined and used.

String

Strings mark a single value per structure and are commonly used to simplify and make complicated values more user-friendly. Below is an example of a string variable definition.

variable "template" {

type = string

default = "01000000-0000-4000-8000-000030080200"

}

A string variable can then be used in resource plans. Surrounded by double quotes, string variables are a simple substitution such as the example underneath.

storage = var.template

List

Another type of Terraform variables lists. They work much like a numbered catalogue of values. Each value can be called by their corresponding index in the list. Here is an example of a list variable definition.

variable "users" {

type = list

default = ["root", "user1", "user2"]

}

Lists can be used in the resource plans similarly to strings, but you'll also need to denote the index of the value you are looking for.

```
username = var.users[0]
```

## Map

Maps are a collection of string keys and string values. These can be useful for selecting values based on predefined parameters such as the server configuration by the monthly price.

```
variable "plans" {  
    type = map  
    default = {  
        "5USD" = "1xCPU-1GB"  
        "10USD" = "1xCPU-2GB"  
        "20USD" = "2xCPU-4GB"  
    }  
}
```

You can access the right value by using the matching key. For example, the variable below would set the plan to "1xCPU-1GB".

```
plan = var.plans["5USD"]
```

The values matching to their keys can also be used to look up information in other maps.

For example, underneath is a shortlist of plans and their corresponding storage sizes.

```
variable "storage_sizes" {  
    type = map  
    default = {  
        "1xCPU-1GB" = "25"  
        "1xCPU-2GB" = "50"  
        "2xCPU-4GB" = "80"  
    }  
}
```

These can then be used to find the right storage size based on the monthly price as defined in the previous example.

```
size = lookup(var.storage_sizes, var.plans["5USD"])
```

## Boolean

The last of the available variable type is boolean. They give the option to employ simple true or false values. For example, you might wish to have a variable that decides when to generate the root user password on a new deployment.

```
variable "set_password" {  
    default = false  
}
```

The above example boolean can be used similarly to a string variable by simply marking down the correct variable.

```
create_password = var.set_password
```

By default, the value is set to false in this example. However, you can overwrite the variable at deployment by assigning a different value in a command-line variable.

```
terraform apply -var set_password="true"
```

### NEW QUESTION: 43

By default, provisioners that fail will also cause the Terraform apply itself to error. How can you change this default behavior within a provisioner?

- A. provisioner "local-exec" { on\_failure = "next" }
- B. provisioner "local-exec" { when = "failure" terraform apply }
- C. provisioner "local-exec" { on\_failure = "continue" }
- D. provisioner "local-exec" { on\_failure = continue }

**Answer:** ([SHOW ANSWER](#))

Explanation

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

### NEW QUESTION: 44

Using multi-cloud and provider-agnostic tools provides which of the following benefits?

- A. Operations teams only need to learn and manage a single tool to manage infrastructure, regardless of where the infrastructure is deployed.
- B. Increased risk due to all infrastructure relying on a single tool for management.
- C. Can be used across major cloud providers and VM hypervisors.
- D. Slower provisioning speed allows the operations team to catch mistakes before they are applied.

**Answer:** ([SHOW ANSWER](#))

Explanation

Using a tool like Terraform can be advantageous for organizations deploying workloads across multiple public and private cloud environments. Operations teams only need to learn a single tool, single language, and can use the same tooling to enable a DevOps-like experience and workflows.

### NEW QUESTION: 45

Which of the following connection types are supported by the remote-exec provisioner?

(select two)

- A. WinRM
- B. UDP
- C. SMB
- D. RDP
- E. ssh

**Answer:** ([SHOW ANSWER](#))

The remote-exec provisioner invokes a script on a remote resource after it is created. The remote-exec provisioner supports both ssh and winrm type connections.

remote-exec connection types -

\* ssh on Linux

\* winrm on Windows

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

### NEW QUESTION: 46

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

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

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

### NEW QUESTION: 47

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address aws\_instance.web refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.
- C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.
- D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer:** ([SHOW ANSWER](#))

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:  
module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

`aws_instance.web[3]`

Refers to only the last instance in the config, and an address like this:

`aws_instance.web`

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

## NEW QUESTION: 48

Which of the following clouds does not have a provider maintained HashiCorp?

- A. IBM Cloud
- B. DigitalOcean
- C. OpenStack
- D. AWS

**Answer: ([SHOW ANSWER](#))**

IBM Cloud does not have a provider maintained by HashiCorp, although IBM Cloud does maintain their own Terraform provider.

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

## NEW QUESTION: 49

```
resource "aws_s3_bucket" "example" { bucket = "my-test-s3-terraform-bucket" ... }  
resource "aws_iam_role"  
"test_role" { name = "test_role" ... }
```

Due to the way that the application code is written , the s3 bucket must be created before the test role is created , otherwise there will be a problem. How can you ensure that?

- A. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- B. Add explicit dependency using `depends_on` . This will ensure the correct order of resource creation.
- C. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.
- D. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.

**Answer: ([SHOW ANSWER](#))**

## Explanation

Use the depends\_on meta-argument to handle hidden resource dependencies that Terraform can't automatically infer.

Explicitly specifying a dependency is only necessary when a resource relies on some other resource's behavior but doesn't access any of that resource's data in its arguments.

## NEW QUESTION: 50

- 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

**Answer:** ([SHOW ANSWER](#))

- \* 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>

## NEW QUESTION: 51

What is the command you can use to set an environment variable named "var1" of type String?

- A. export TF\_VAR\_VAR1
- B. set TF\_VAR\_var1
- C. variable "var1" { type = "string"}
- D. export TF\_VAR\_var1

**Answer:** ([SHOW ANSWER](#))

The environment variable must be in the format TF\_VAR\_name, so for the

## NEW QUESTION: 52

Terraform works well in Windows but a Windows server is required.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 53

lookup retrieves the value of a single element from which of the below data type?

- A. map
- B. set
- C. string
- D. list

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

### NEW QUESTION: 54

Environment variables can be used to set variables. The environment variables must be in the format

"\_\_\_\_\_"\_<variablename>. Select the correct prefix string from the following list.

- A. TF\_CLI\_ARGS
- B. TF\_VAR
- C. TF\_VAR\_
- D. TF\_VAR\_ENV

**Answer:** C ([LEAVE A REPLY](#))

Explanation

Environment variables can be used to set variables. The environment variables must be in the format TF\_VAR\_name and this will be checked last for a value. For example:

```
export TF_VAR_region=us-west-1
```

```
export TF_VAR_ami=ami-049d8641
```

```
export TF_VAR_alist='[1,2,3]'
```

```
export TF_VAR_amap='{ foo = "bar", baz = "qux" }'
```

<https://www.terraform.io/docs/commands/environment-variables.html>

### NEW QUESTION: 55

Terraform import command can import resources into modules as well directly into the root of your state.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

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 directly into the root of your state.

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform. For example:

```
resource "aws_instance" "import_example" {  
  # ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration:

```
$ terraform import aws_instance.import_example i-03efafa258104165f
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...
aws_instance.import_example: Import complete!
Imported aws_instance (ID: i-03efafa258104165f)
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import
successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

As a result of the above command, the resource is recorded in the state file. We can now run terraform plan to see how the configuration compares to the imported resource, and make any adjustments to the configuration to align with the current (or desired) state of the imported object.

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

## NEW QUESTION: 56

When you initialize Terraform, where does it cache modules from the public Terraform Module Registry?

- A. On disk in the /tmp directory
- B. In memory
- C. On disk in the .terraform sub-directory
- D. They are not cached

**Answer: ([SHOW ANSWER](#))**

## NEW QUESTION: 57

You have created a custom variable definition file my\_vars.tfvars. How will you use it for provisioning infrastructure?

- A. `terraform apply -var-state-file ="my_vars.tfvars"`
- B. `terraform apply var-file="my_vars.tfvars"`
- C. `terraform plan -var-file="my_vars.tfvar"`
- D. `terraform apply -var-file="my_vars.tfvars"`

**Answer: D ([LEAVE A REPLY](#))**

To set lots of variables, it is more convenient to specify their values in a variable definitions file (with a filename ending in either .tfvars or .tfvars.json) and then specify that file on the command line with -var-file:

`terraform apply -var-file="my_vars.tfvars"`

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

## NEW QUESTION: 58

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

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

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

## NEW QUESTION: 59

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 taint command targeting the VMs then run Terraform plan and Terraform apply
- B. Delete the Terraform VM resources from your Terraform code then run Terraform plan and terraform apply
- C. Use the terraform state rm command to remove the VM from state file
- D. Use the terraform apply command targeting the VM resources only

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 60

- A. export TF\_VAR\_VAR1
- B. set TF\_VAR\_var1
- C. variable "var1" { type = "string"}
- D. export TF\_VAR\_var1

**Answer:** D ([LEAVE A REPLY](#))

The environment variable must be in the format TF\_VAR\_name, so for the

[https://www.terraform.io/docs/commands/environment-variables.html#tf\\_var\\_name](https://www.terraform.io/docs/commands/environment-variables.html#tf_var_name)

## NEW QUESTION: 61

All modules published on the official Terraform Module Registry have been verified by HashiCorp.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!  
Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com  
**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 62

terraform init retrieves the source code tot all referenced modules

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

## NEW QUESTION: 63

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

- A. format
- B. join
- C. tostring

Explanation

<https://www.terraform.io/docs/configuration/functions/tostring.html>

- D. replace

**Answer: ([SHOW ANSWER](#))**

## NEW QUESTION: 64

Which of the following terraform subcommands could be used to remove the lock on the state for the current configuration?

- A. Unlock
- B. force-unlock
- C. Removing the lock on a state file is not possible
- D. state-unlock

**Answer: ([SHOW ANSWER](#))**

Explanation

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

## NEW QUESTION: 65

Terraform validate reports syntax check errors from which of the following scenarios?

- A. The state files does not match the current infrastructure
- B. Code contains tabs indentation instead of spaces
- C. None of the above
- D. There is missing value for a variable

**Answer: ([SHOW ANSWER](#))**

## NEW QUESTION: 66

Which option can not be used to keep secrets out of Terraform configuration files?

- A. Environment variables
- B. secure string
- C. A Terraform provider
- D. A -var flag

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 67**

What is the result of the following terraform function call?

- A. False

Explanation

<https://www.terraform.io/docs/configuration/functions/index.html>

- B. True

**Answer:** A ([LEAVE A REPLY](#))

### **NEW QUESTION: 68**

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. Collection functions
- B. Local values
- C. Count arguments
- D. Dynamic blocks

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 69**

Terraform plan updates your state file.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 70**

- A. `terraform taint module.foo.module.bar.baz`
- B. `terraform taint module.foo.bar.aws_instance.baz`
- C. `terraform taint module.foo.module.bar.aws_instance.baz`
- D. `terraform taint foo.bar.aws_instance.baz`

**Answer:** ([SHOW ANSWER](#))

Explanation

Check resource addressing

<https://www.terraform.io/docs/internals/resource-addressing.html>

### **NEW QUESTION: 71**

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

**Answer: (SHOW ANSWER)**

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

### **NEW QUESTION: 72**

When TF\_LOG\_PATH is set, TF\_LOG must be set in order for any logging to be enabled.

- A. False
- B. True

**Answer: B (LEAVE A REPLY)**

Explanation

TF\_LOG\_PATH specifies where the log should persist its output to. Note that even when TF\_LOG\_PATH is set, TF\_LOG must be set in order for any logging to be enabled.

For example, to always write the log to the directory you're currently running terraform from:

```
export TF_LOG_PATH=./terraform.log  
export TF_LOG=TRACE
```

### **NEW QUESTION: 73**

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

**Answer: C (LEAVE A REPLY)**

terraform workspace list

The command will list all existing workspaces.

Reference: <https://www.terraform.io/docs/cli/commands/workspace/list.html>

### **NEW QUESTION: 74**

When configuring a remote backend in Terraform, it might be a good idea to purposely omit some of the required arguments to ensure secrets and other important data aren't inadvertently shared with others. What are the ways the remaining configuration can be added to Terraform so it can initialize and communicate with the backend? (select three)

- A. interactively on the command line

Explanation

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process. There are several ways to supply the remaining arguments: <https://www.terraform.io/docs/backends/init.html#backend-initialization>

- B. command-line key/value pairs
- C. use the -backend-config=PATH to specify a separate config file
- D. directly querying HashiCorp Vault for the secrets

Answer: ([SHOW ANSWER](#))

## NEW QUESTION: 75

Which one is the right way to import a local module names consul?

- A. module "consul" { source = "consul"}
- B. module "consul" { source = "./consul"}
- C. module "consul" { source = "../consul"}
- D. module "consul" { source = "module/consul"}

Answer: ([SHOW ANSWER](#))

Explanation

A local path must begin with either ./ or ../ to indicate that a local path is intended, to distinguish from a module registry address.

```
module "consul" {
  source = "./consul"
}
```

## NEW QUESTION: 76

When does Sentinel enforce policy logic during a Terraform Enterprise run?

- A. After the apply phase
- B. During the plan phase
- C. Before the a apply phase
- D. Before the plan phase

Answer: ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

### **NEW QUESTION: 77**

- Why might a user opt to include the following snippet in their configuration file?
- A. this ensures that all Terraform providers are above a certain version to match the application being deployed
  - B. versions before Terraform 0.12 were not approved by HashiCorp to be used in production
  - C. Terraform 0.12 introduced substantial changes to the syntax used to write Terraform configuration
  - D. The user wants to ensure that the application being deployed is a minimum version of 0.12

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 78**

You have to initialize a Terraform backend before it can be configured.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 79**

ABC Enterprise has recently tied up with multiple small organizations for exchanging database information. Due to this, the firewall rules are increasing and are more than 100 rules. This is leading firewall configuration file that is difficult to manage. What is the way this type of configuration can be managed easily?

- A. Terraform Functions
- B. Terraform Expression
- C. Terraform Backends
- D. Dynamic Blocks

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 80**

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

## Explanation

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

### NEW QUESTION: 81

When using constraint expressions to signify a version of a provider, which of the following are valid provider versions that satisfy the expression found in the following code snippet:

(select two)

1. terraform
  2. {
  3. required\_providers
  4. {
  5. aws = "~> 1.2.0"
  6. }
  7. }
- A. 1.3.1  
B. 1.2.3  
C. 1.2.9  
D. 1.3.0

**Answer: B,C** ([LEAVE A REPLY](#))

## Explanation

As your Terraform usage becomes more advanced, there are some cases where you may need to modify the Terraform state. Rather than modify the state directly, the `terraform state` commands can be used in many cases instead. This command is a nested subcommand, meaning that it has further subcommands.

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

### NEW QUESTION: 82

Which of the following can you do with `terraform plan`? Choose two correct answers.

- A. Save a generated execution plan to apply later  
B. View the execution plan and check if the changes match your expectations  
C. Schedule Terraform to run at a planned time in the future  
D. Execute a plan in a different workspace

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 83

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 plan 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

**Answer:** ([SHOW ANSWER](#))

changes, and terraform apply to create new infrastructure.

Reference:

+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply  
+to+create+new  
+infrastructure.&oq=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run  
+terraform+plan+to  
+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new  
+infrastructure.&aqs=chrome..69i57.556j0j7&sourceid=chrome&ie=UTF-8

## NEW QUESTION: 84

Where does the Terraform local backend store its state?

- A.** In the /tmp directory
- B.** In the terraform.tfvars file
- C.** In the terraform.tfstate file
- D.** In the user's .terraformrc file

**Answer:** ([SHOW ANSWER](#))

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>

## NEW QUESTION: 85

Terraform providers are always installed from the Internet.

- A.** True
- B.** False

**Answer:** ([SHOW ANSWER](#))

Explanation

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>

## NEW QUESTION: 86

You have a Terraform configuration file where a variable itemNum is defined as follows:

```
variable "itemNum" { default = 3}
```

You also have a defined the following environment variables in your shell: TF\_itemNum =6, TF\_VAR\_itemNum =9. You also have a terraform.tfvars file with the following contents

itemNum = 7 When you run the following apply command, what is the value assigned to the itemNum variable?

terraform apply -var itemNum =4

- A. 10
- B. 6
- C. 1
- D. 4
- E. 3

**Answer:** ([SHOW ANSWER](#))

Explanation

The -var and -var-file methods of assigning variables have the highest precedence.

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

### **NEW QUESTION: 87**

As a member of the operations team, you need to run a script on a virtual machine created by Terraform. Which provisioner is best to use in your Terraform code?

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

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 88**

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 not make any changes
- B. Terraform will remove the VM from state file
- C. Terraform will report an error
- D. Terraform will recreate the VM

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 89**

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. Terraform attempts to provision the resource up to three times before exiting with an error
- B. the terraform plan is rolled back and all provisioned resources are removed
- C. it is automatically deleted
- D. the resource is marked as tainted

**Answer: ([SHOW ANSWER](#))**

Explanation

If a resource successfully creates but fails during provisioning, Terraform will error and mark the resource as

"tainted". A resource that is tainted has been physically created, but can't be considered safe to use since provisioning failed. Terraform also does not automatically roll back and destroy the resource during the apply when the failure happens, because that would go against the execution plan: the execution plan would've said a resource will be created, but does not say it will ever be deleted.

**NEW QUESTION: 90**

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

Provisioners are a Last Resort

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.

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

**NEW QUESTION: 91**

Terraform-specific settings and behaviors are declared in which configuration block type?

- A. provider
- B. terraform
- C. resource
- D. data

**Answer: ([SHOW ANSWER](#))**

The special terraform configuration block type is used to configure some behaviors of Terraform itself, such as requiring a minimum Terraform version to apply your configuration.

Example

```
terraform {  
  required_version = "> 0.12.0"  
}
```

<https://www.terraform.io/docs/configuration/terraform.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (**384 Q&As Dumps, 30%OFF**  
**Special Discount: freecram**)

## NEW QUESTION: 92

Which one of the following will run echo 0 and echo 1 on a newly created host?

A. provisioner "local-exec" { command = "echo 0"

command = "echo 1"

}

B. provisioner "remote-exec" {

inline = [

echo 0,

echo 1

]

}

C. provisioner "remote-exec" {

command = "\${echo 0}"

command = "\${echo 1}"

}

D. provisioner "remote-exec" {

inline = [

"echo 0",

"echo 1"

]

}

**Answer:** ([SHOW ANSWER](#))

Explanation

remote-exec Provisioner

Example usage

```
resource "aws_instance" "web" {  
# ...  
provisioner "remote-exec" {  
inline = [  
"puppet apply",  
"consul join ${aws_instance.web.private_ip}",  
]  
}  
}
```

### **NEW QUESTION: 93**

Which of the following is not true of Terraform providers?

- A. Providers can be maintained by a community of users
- B. Providers can be written by individuals
- C. None of the above
- D. Major cloud vendors and non-cloud vendors can write, maintain, or collaborate on Terraform providers
- E. Some providers are maintained by HashiCorp

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 94**

If a module declares a variable with a default, that variable must also be defined within the module.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 95**

Given the Terraform configuration below, in which order will the resources be created?

- A. Larger image
- B. resources will be created simultaneously
- C. aws\_eip will be created first aws\_instance will be created second
- D. aws\_instance will be created first aws\_eip will be created second

**Answer:** ([SHOW ANSWER](#))

Explanation

The aws\_instance will be created first, and then aws\_eip will be created second due to the aws\_eip's resource dependency of the aws\_instance id

### **NEW QUESTION: 96**

resource "aws\_s3\_bucket" "example" { bucket = "my-test-s3-terraform-bucket" ...} resource "aws\_iam\_role" "test\_role" { name = "test\_role" ...} Due to the way that the application code is written , the s3 bucket must be created before the test role is created , otherwise there will be a problem. How can you ensure that?

- A. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- B. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- C. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.
- D. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.

**Answer:** ([SHOW ANSWER](#))

Use the depends\_on meta-argument to handle hidden resource dependencies that Terraform can't automatically infer.

Explicitly specifying a dependency is only necessary when a resource relies on some other resource's behavior but doesn't access any of that resource's data in its arguments.

### **NEW QUESTION: 97**

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

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

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 98**

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

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 99**

Terra form installs its providers during which phase?

- A. Init
- B. All of the above
- C. Man
- D. Refresh

**Answer: A** ([LEAVE A REPLY](#))

## NEW QUESTION: 100

What is the name assigned by Terraform to reference this resource?

```
mainresource "google_compute_instance" "main" {  
    name = "test"  
}
```

- A. google
- B. test
- C. compute\_instance
- D. main

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 101

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

**Answer:** ([SHOW ANSWER](#))

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

## NEW QUESTION: 102

When using Terraform in a team it is important for everyone to be working with the same state so that operations will be applied to the same remote objects. Which of the below option is a recommended solution for this?

- A. Remote State
- B. Module
- C. Use the cached state and treat this as the record of truth.
- D. Workspace

**Answer:** ([SHOW ANSWER](#))

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

## NEW QUESTION: 103

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

**Answer: ([SHOW ANSWER](#))**

The terraform import command is used to import existing infrastructure. Import existing Google Cloud resources into Terraform with Terraformer.

#### **NEW QUESTION: 104**

How is terraform import run?

- A. As a part of terraform refresh
- B. As a part of terraform plan
- C. As a part of terraform init
- D. All of the above
- E. By an explicit call

**Answer: B ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 105**

terraform refresh command will not modify infrastructure, but does modify the state file.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

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.

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

#### **NEW QUESTION: 106**

Refer to the below code where developer is outputting the value of the database password but has used sensitive parameter to hide the output value in the CLI.

```
output "db_password" { value = aws_db_instance.db.password description = "The password for logging in to the database." sensitive = true}
```

Since sensitive is set to true, the value associated with db password will not be present in state file as plain-text?

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

Explanation

Sensitive output values are still recorded in the state, and so will be visible to anyone who is able to access the state data.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

## NEW QUESTION: 107

Which of the following best describes the default local backend?

- A. The local backend is where Terraform Enterprise stores logs to be processed by an log collector.
- B. The local backend stores state on the local filesystem, locks the state using system APIs, and performs operations locally.
- C. The local backend is the directory where resources deployed by Terraform have direct access to in order to update their current state.
- D. The local backend is how Terraform connects to public cloud services, such as AWS, Azure, or GCP.

**Answer:** ([SHOW ANSWER](#))

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

```
terraform {  
  backend "local" {  
    path = "relative/path/to/terraform.tfstate"  
  }  
}
```

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

## NEW QUESTION: 108

True or False: A list(...) contain a number of values of the same type while an object(...) can contain a number of values of different types.

- A. False
- B. True

**Answer:** ([SHOW ANSWER](#))

Collection Types

A collection type allows multiple values of one other type to be grouped together as a single value. The type of value within a collection is called its element type. All collection types must have an element type, which is provided as the argument to their constructor. For example, the type list(string) means "list of strings", which is a different type than list(number), a list of numbers. All elements of a collection must always be of the same type.

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.

\* `set(...)`: a collection of unique values that do not have any secondary identifiers or ordering.

<https://www.terraform.io/docs/configuration/types.html>

## Structural Types

A structural type allows multiple values of several distinct types to be grouped together as a single value. Structural types require a schema as an argument, to specify which types are allowed for which elements.

The two kinds of structural type in the Terraform language are:

\* `object(...)`: a collection of named attributes that each have their own type.

The schema for object types is `{ <KEY> = <TYPE>, <KEY> = <TYPE>, ... }` - a pair of curly braces containing a comma-separated series of `<KEY> = <TYPE>` pairs. Values that match the object type must contain all of the specified keys, and the value for each key must match its specified type. (Values with additional keys can still match an object type, but the extra attributes are discarded during type conversion.)

\* `tuple(...)`: a sequence of elements identified by consecutive whole numbers starting with zero, where each element has its own type.

The schema for tuple types is `[<TYPE>, <TYPE>, ...]` - a pair of square brackets containing a comma-separated series of types. Values that match the tuple type must have exactly the same number of elements (no more and no fewer), and the value in each position must match the specified type for that position.

For example: an object type of `object({ name=string, age=number })` would match a value like the following:

```
{  
  name = "John"  
  age = 52  
}
```

Also, an object type of `object({ id=string, cidr_block=string })` would match the object produced by a reference to an `aws_vpc` resource, like `aws_vpc.example_vpc`; although the resource has additional attributes, they would be discarded during type conversion.

Finally, a tuple type of `tuple([string, number, bool])` would match a value like the following:  
["a", 15, true]

<https://www.terraform.io/docs/configuration/types.html>

## **NEW QUESTION: 109**

Which task does terraform init not perform?

- 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

**Answer: D ([LEAVE A REPLY](#))**

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

## **NEW QUESTION: 110**

- A. The instance type will be changed to t2.micro and again will be changed to t2.medium
- B. No resource will be updated and you will see the message : Apply Complete !  
Resources : 0 added, 0 changed, 0 destroyed.
- C. 1 resource will be updated and you will see the message : Apply Complete !  
Resources : 0 added, 1 changed, 0 destroyed.
- D. terraform apply will through an error.

**Answer: ([SHOW ANSWER](#))**

## **NEW QUESTION: 111**

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

- A. terraform providers
- B. terraform state show
- C. terraform provider
- D. terraform plan

**Answer: ([SHOW ANSWER](#))**

Use the terraform providers command to view the specified version constraints for all providers used in the current configuration.

<https://www.terraform.io/docs/configuration/providers.html>

## **NEW QUESTION: 112**

State locking does not happen automatically and must be specified at run

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

Explanation

State locking happens automatically on all operations that could write state.

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

## **NEW QUESTION: 113**

Every region in AWS has a different AMI ID for Linux and these are keep on changing.

What is the best approach to create the EC2 instances that can deal with different AMI IDs based on regions?

- A. Use data source aws\_ami.
- B. Create a map of region to ami id.
- C. Create different configuration file for different region.
- D. None of the above

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/data-sources.html>

## **NEW QUESTION: 114**

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

**Answer: ([SHOW ANSWER](#))**

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.

## **NEW QUESTION: 115**

Select all Operating Systems that Terraform is available for. (select five)

- A. Linux
- B. macOS
- C. Unix
- D. Solaris
- E. Windows
- F. FreeBSD

**Answer: ([SHOW ANSWER](#))**

Terraform is available for macOS, FreeBSD, OpenBSD, Linux, Solaris, Windows

<https://www.terraform.io/downloads.html>

## **NEW QUESTION: 116**

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

- A. terraform providers
- B. terraform state show
- C. terraform provider
- D. terraform plan

**Answer: ([SHOW ANSWER](#))**

Explanation

Use the terraform providers command to view the specified version constraints for all providers used in the current configuration.

<https://www.terraform.io/docs/configuration/providers.html>

### **NEW QUESTION: 117**

If a module uses a local variable, you can expose that value with a terraform output.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

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>

### **NEW QUESTION: 118**

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 plan 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.

**Answer: ([SHOW ANSWER](#))**

Explanation/Reference: <https://www.google.com/search?q=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new+infrastructure.&oq=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new+infrastructure.&aqs=chrome..69i57.556j0j7&sourceid=chrome&ie=UTF-8>

## NEW QUESTION: 119

You want terraform plan and apply to be executed in Terraform Cloud's run environment but the output is to be streamed locally. Which one of the below you will choose?

- A. Local Backends
- B. This can be done using any of the local or remote backends
- C. Remote Backends
- D. Terraform Backends

**Answer:** ([SHOW ANSWER](#))

Explanation

The remote backend stores Terraform state and may be used to run operations in Terraform Cloud.

When using full remote operations, operations like terraform plan or terraform apply can be executed in Terraform Cloud's run environment, with log output streaming to the local terminal.

Remote plans and applies use variable values from the associated Terraform Cloud workspace.

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

## NEW QUESTION: 120

- A. Run terraform state list to find the name of the resource, then terraform state show to find the attributes including public IP address
- 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 output ip\_address to view the result
- D. Run terraform destroy then terraform apply and look for the IP address in stdout

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 121

In the example below, the depends\_on argument creates what type of dependency?

- A. implicit dependency
- B. internal dependency
- C. non-dependency resource
- D. explicit dependency

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!  
Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com  
**TA-002-P exam questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30% OFF**

**Special Discount: freecram**)

### **NEW QUESTION: 122**

As a developer, you want to ensure your plugins are up to date with the latest versions.

Which Terraform command should you use?

- A. terreform providers- upgrade
- B. terreform refresh -upgrade
- C. terreform apply -upgrade
- D. terreformn Init -upgrade

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 123**

Terraform-specific settings and behaviors are declared in which configuration block type?

- A. provider
- B. terraform
- C. resource
- D. data

**Answer:** ([SHOW ANSWER](#))

Explanation

The special terraform configuration block type is used to configure some behaviors of Terraform itself, such as requiring a minimum Terraform version to apply your configuration.

Example

```
terraform {  
  required_version = "> 0.12.0"  
}
```

<https://www.terraform.io/docs/configuration/terraform.html>

### **NEW QUESTION: 124**

Complete the following sentence:

For local state, the workspaces are stored directly in a \_\_\_\_\_.

- A. a file called terraform.tfstate.backup
- B. directory called terraform.workspaces.tfstate
- C. a file called terraform.tfstate
- D. directory called terraform.tfstate.d

**Answer:** ([SHOW ANSWER](#))

For local state, Terraform stores the workspace states in a directory called terraform.tfstate.d.

<https://www.terraform.io/docs/state/workspaces.html#workspace-internals>

## **NEW QUESTION: 125**

Terraform will sync all resources in state by default for every plan and apply, hence for larger infrastructures this can slow down terraform plan and terraform apply commands?

A. False

B. True

**Answer: B ([LEAVE A REPLY](#))**

For small infrastructures, Terraform can query your providers and sync the latest attributes from all your resources. This is the default behavior of Terraform: for every plan and apply, Terraform will sync all resources in your state.

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to query multiple resources at once, and the round trip time for each resource is hundreds of milliseconds. On top of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

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

## **NEW QUESTION: 126**

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.

**Answer: ([SHOW ANSWER](#))**

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

## **NEW QUESTION: 127**

You have a Terraform configuration file where a variable `itemNum` is defined as follows:  
`variable "itemNum" { default = 3}`

You also have defined the following environment variables in your shell: `TF_itemNum =6`, `TF_VAR_itemNum =9`. You also have a `terraform.tfvars` file with the following contents  
`itemNum = 7` When you run the following apply command, what is the value assigned to the `itemNum` variable?

`terraform apply -var itemNum =4`

- A. 10
- B. 6
- C. 1
- D. 4
- E. 3

**Answer:** ([SHOW ANSWER](#))

The -var and -var-file methods of assigning variables have the highest precedence.

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

### **NEW QUESTION: 128**

Your organization has moved to AWS and has manually deployed infrastructure using the console. Recently, a decision has been made to standardize on Terraform for all deployments moving forward.

What can you do to ensure that all existing is managed by Terraform moving forward without interruption to existing services?

- A. Submit a ticket to AWS and ask them to export the state of all existing resources and use terraform import to import them into the state file.
- B. Delete the existing resources and recreate them using new a Terraform configuration so Terraform can manage them moving forward.
- C. Resources that are manually deployed in the AWS console cannot be imported by Terraform.
- D. Using terraform import, import the existing infrastructure into your Terraform state.

**Answer:** ([SHOW ANSWER](#))

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
  # ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import  
successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

### **NEW QUESTION: 129**

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

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 130**

- A. The module must be PCI/HIPPA compliant.
- B. Module repositories must use this three-part name format, terraform-- .
- C. The registry uses tags to identify module versions.
- D. Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .
- E. The module must be on GitHub and must be a public repo.

**Answer:** ([SHOW ANSWER](#))

Explanation

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

### **NEW QUESTION: 131**

Anyone can publish and share modules on the Terraform Public Module Registry, and meeting the requirements for publishing a module is extremely easy. Select from the following list all valid requirements.

(select three)

- A. The module must be PCI/HIPPA compliant.
- B. Module repositories must use this three-part name format, terraform-- .
- C. The registry uses tags to identify module versions.
- D. Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .
- E. The module must be on GitHub and must be a public repo.

**Answer:** ([SHOW ANSWER](#))

Explanation

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

## **NEW QUESTION: 132**

Terraform must track metadata such as resource dependencies. Where is this data stored?

- A.** workspace
- B.** backend
- C.** state file
- D.** metadata store

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

<https://www.terraform.io/docs/state/purpose.html#metadata>

## **NEW QUESTION: 133**

You want to use terraform import to start managing infrastructure that was not originally provisioned through infrastructure as code. Before you can import the resource's current state, what must you do in order to prepare to manage these resources using Terraform?

- A.** Run terraform refresh to ensure that the state file has the latest information for existing resources.
- B.** Update the configuration file to include the new resources.
- C.** Shut down or stop using the resources being imported so no changes are inadvertently missed.
- D.** Modify the Terraform state file to add the new resources.

**Answer:** ([SHOW ANSWER](#))

Explanation

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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. The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

### **NEW QUESTION: 134**

What feature of Terraform Cloud and/or Terraform Enterprise can you publish and maintain a set of custom modules which can be used within your organization?

- A. custom VCS integration
- B. remote runs
- C. Terraform registry
- D. private module registry

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 135**

Which of the below configuration file formats are supported by Terraform? (Select TWO)

- A. Node
- B. JSON
- C. Go
- D. YAML
- E. HCL

**Answer: ([SHOW ANSWER](#))**

Terraform supports both HashiCorp Configuration Language (HCL) and JSON formats for configurations.

<https://www.terraform.io/docs/configuration/>

### **NEW QUESTION: 136**

After executing a terraform apply, you notice that a resource has a tilde (~) next to it. What does this infer?

- A. The resource will be updated in place.
- B. The resource will be created.
- C. Terraform can't determine how to proceed due to a problem with the state file.
- D. The resource will be destroyed and recreated.

**Answer:** ([SHOW ANSWER](#))

Explanation

The prefix -/+ means that Terraform will destroy and recreate the resource, rather than updating it in-place.

The prefix ~ means that some attributes and resources can be updated in-place.

\$ terraform apply

```
aws_instance.example: Refreshing state... [id=i-0bbf06244e44211d1]
```

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

-/+ destroy and then create replacement

Terraform will perform the following actions:

# aws\_instance.example must be replaced

-/+ resource "aws\_instance" "example" {

~ ami = "ami-2757f631" -> "ami-b374d5a5" # forces replacement

~ arn = "arn:aws:ec2:us-east-1:130490850807:instance/i-0bbf06244e44211d1" -> (known after apply)

~ associate\_public\_ip\_address = true -> (known after apply)

~ availability\_zone = "us-east-1c" -> (known after apply)

~ cpu\_core\_count = 1 -> (known after apply)

~ cpu\_threads\_per\_core = 1 -> (known after apply)

- disable\_api\_termination = false -> null

- ebs\_optimized = false -> null

get\_password\_data = false

+ host\_id = (known after apply)

~ id = "i-0bbf06244e44211d1" -> (known after apply)

~ instance\_state = "running" -> (known after apply)

instance\_type = "t2.micro"

~ ipv6\_address\_count = 0 -> (known after apply)

~ ipv6\_addresses = [] -> (known after apply)

+ key\_name = (known after apply)

- monitoring = false -> null

+ network\_interface\_id = (known after apply)

+ password\_data = (known after apply)

+ placement\_group = (known after apply)

~ primary\_network\_interface\_id = "eni-0f1ce5bdae258b015" -> (known after apply)

```
~ private_dns = "ip-172-31-61-141.ec2.internal" -> (known after apply)
~ private_ip = "172.31.61.141" -> (known after apply)
~ public_dns = "ec2-54-166-19-244.compute-1.amazonaws.com" -> (known after apply)
~ public_ip = "54.166.19.244" -> (known after apply)
~ security_groups = [
- "default",
] -> (known after apply)
source_dest_check = true
~ subnet_id = "subnet-1facdf35" -> (known after apply)
~ tenancy = "default" -> (known after apply)
~ volume_tags = {} -> (known after apply)
~ vpc_security_group_ids = [
- "sg-5255f429",
] -> (known after apply)
- credit_specification {
- cpu_credits = "standard" -> null
}
+ ebs_block_device {
+ delete_on_termination = (known after apply)
+ device_name = (known after apply)
+ encrypted = (known after apply)
+ iops = (known after apply)
+ snapshot_id = (known after apply)
+ volume_id = (known after apply)
+ volume_size = (known after apply)
+ volume_type = (known after apply)
}
+ ephemeral_block_device {
+ device_name = (known after apply)
+ no_device = (known after apply)
+ virtual_name = (known after apply)
}
+ network_interface {
+ delete_on_termination = (known after apply)
+ device_index = (known after apply)
+ network_interface_id = (known after apply)
}
~ root_block_device {
~ delete_on_termination = true -> (known after apply)
~ iops = 100 -> (known after apply)
~ volume_id = "vol-0079e485d9e28a8e5" -> (known after apply)
```

```
~ volume_size = 8 -> (known after apply)
~ volume_type = "gp2" -> (known after apply)
}
}

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

### NEW QUESTION: 137

Environment variables can be used to set variables. The environment variables must be in the format "      "\_<variablename>. Select the correct prefix string from the following list.

- A. TF\_CLI\_ARGS
- B. TF\_VAR
- C. TF\_VAR\_
- D. TF\_VAR\_ENV

**Answer:** ([SHOW ANSWER](#))

Environment variables can be used to set variables. The environment variables must be in the format TF\_VAR\_name and this will be checked last for a value. For example:

```
export TF_VAR_region=us-west-1
export TF_VAR_ami=ami-049d8641
export TF_VAR_alist='[1,2,3]'
export TF_VAR_amap='{ foo = "bar", baz = "qux" }'
```

<https://www.terraform.io/docs/commands/environment-variables.html>

### NEW QUESTION: 138

In regards to deploying resources in multi-cloud environments, what are some of the benefits of using Terraform rather than a provider's native tooling? (select three)

- A. Terraform can help businesses deploy applications on multiple clouds and on-premises infrastructure.
- B. Terraform is not cloud-agnostic and can be used to deploy resources across a single public cloud.
- C. Terraform simplifies management and orchestration, helping operators build large-scale, multi-cloud infrastructure.
- D. Terraform can manage cross-cloud dependencies.

**Answer: ([SHOW ANSWER](#))**

Terraform is cloud-agnostic and allows a single configuration to be used to manage multiple providers, and to even handle cross-cloud dependencies. This simplifies management and orchestration, helping operators build large-scale multi-cloud infrastructures.

<https://www.terraform.io/intro/use-cases.html>

**NEW QUESTION: 139**

What does terraform refresh command do?

- A. terraform refresh command basically updates the configuration file with the current state of the actual infrastructure
- B. terraform refresh is use to change/modify the infrastructure based on the existing state file, at that moment.
- C. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- D. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- E. terraform refresh syncs the state file with the real world infrastructure.

**Answer: ([SHOW ANSWER](#))****NEW QUESTION: 140**

- A. Full API Coverage
- B. Secure variable Storage
- C. Roles/ Team management
- D. Cost Estimation
- E. Private Module Registry
- F. Sentinel policies

**Answer: ([SHOW ANSWER](#))**

<https://www.hashicorp.com/products/terraform/pricing/>

**NEW QUESTION: 141**

Terraform works well in Windows but a Windows server is required.

- A. False
- B. True

**Answer: ([SHOW ANSWER](#))**

Explanation

You may see this question in actual exam. Please remember : Terraform does not require GO language to be installed as a prerequisite and it does not require a Windows Server as well.

**NEW QUESTION: 142**

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A. TF\_TRACE
- B. TF\_DEBUG
- C. TF\_LOG
- D. TF\_INFO

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

### NEW QUESTION: 143

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. it is automatically deleted
- B. the terraform plan is rolled back and all provisioned resources are removed
- C. Terraform attempts to provision the resource up to three times before exiting with an error
- D. the resource is marked as tainted

Explanation

If a resource successfully creates but fails during provisioning, Terraform will error and mark the resource as "tainted". A resource that is tainted has been physically created, but can't be considered safe to use since provisioning failed. Terraform also does not automatically roll back and destroy the resource during the apply when the failure happens, because that would go against the execution plan: the execution plan would've said a resource will be created, but does not say it will ever be deleted.

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 144

Given the Terraform configuration below, in which order will the resources be created?

- A. Larger image
- B. resources will be created simultaneously
- C. aws\_eip will be created first aws\_instance will be created second
- D. aws\_instance will be created first aws\_eip will be created second

**Answer:** ([SHOW ANSWER](#))

The aws\_instance will be created first, and then aws\_eip will be created second due to the aws\_eip's resource dependency of the aws\_instance id

#### **NEW QUESTION: 145**

Which of the following is considered a Terraform plugin?

- A. Terraform language
- B. Terraform tooling
- C. Terraform logic
- D. Terraform provider

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

<https://www.terraform.io/docs/plugins/basics.html>

#### **NEW QUESTION: 146**

What is the purpose of a Terraform workspace in either open source or enterprise?

- A. Provides limited access to a cloud environment
- B. A logical separation of business units
- C. Workspaces allow you to manage collections of infrastructure in state files.
- D. A method of grouping multiple infrastructure security policies

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 147**

From the code below, identify the implicit dependency:

- A. The EIP with an id of ami-2757f631
- B. The AMI used for the EC2 instance
- C. The S3 bucket labeled company\_data
- D. The EC2 instance labeled web\_server

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 148**

You are using a terraform operation that writes state. Unfortunately automatic state unlocking has failed for that operation. Which of the below commands can be used to remove the already acquired lock on the state?

- A. terraform unlock
- B. terraform force-unlock
- C. terraform state unlock
- D. None of the above

**Answer: ([SHOW ANSWER](#))**

Explanation

Command: force-unlock

Manually unlock the state for the defined configuration.

This will not modify your infrastructure. This command removes the lock on the state for the current configuration. The behavior of this lock is dependent on the backend being used. Local state files cannot be unlocked by another process.

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

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

Terraform has a force-unlock command to manually unlock the state if unlocking failed.

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.

**NEW QUESTION: 149**

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

- 1. max
- 2. min
- 3. join
- 4. replace
- 5. list
- 6. length
- 7. range

- A. Backends
- B. Data sources
- C. Named values
- D. Functions

**Answer: ([SHOW ANSWER](#))**

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not support user-defined functions, and only the functions built into the language are available for use.

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

**NEW QUESTION: 150**

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:

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. alias = aws.west
  - C. provider = west
  - D. provider = aws.west
- Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 151

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. aws\_instance.web[1].name
- B. aws\_instance.web[2].name
- C. aws\_instance.web[1]
- D. element(aws\_instance.web, 2)
- E. aws\_instance.web.\*.name

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

### NEW QUESTION: 152

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

**Answer:** ([SHOW ANSWER](#))

### NEW QUESTION: 153

Which of the below features of Terraform can be used for managing small differences between different environments which can act more like completely separate working directories.

- A. Repositories
- B. Workspaces
- C. Environment Variables
- D. Backends

**Answer:** ([SHOW ANSWER](#))

workspaces allow conveniently switching between multiple instances of a single configuration within its single backend. They are convenient in a number of situations, but cannot solve all problems.

A common use for multiple workspaces is to create a parallel, distinct copy of a set of infrastructure in order to test a set of changes before modifying the main production infrastructure. For example, a developer working on a complex set of infrastructure changes might create a new temporary workspace in order to freely experiment with changes without affecting the default workspace.

Non-default workspaces are often related to feature branches in version control. The default workspace might correspond to the "master" or "trunk" branch, which describes the intended state of production infrastructure. When a feature branch is created to develop a change, the developer of that feature might create a corresponding workspace and deploy into it a temporary "copy" of the main infrastructure so that changes can be tested without affecting the production infrastructure. Once the change is merged and deployed to the default workspace, the test infrastructure can be destroyed and the temporary workspace deleted.

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

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

#### **NEW QUESTION: 154**

In order to reduce the time it takes to provision resources, Terraform uses parallelism. By default, how many resources will Terraform provision concurrently?

- A. 50
- B. 10
- C. 20
- D. 5

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 155**

True or False. The `terraform refresh` command is used to reconcile the state Terraform knows about (via its state file) with the real-world infrastructure. If drift is detected between the real-world infrastructure and the last known-state, it will modify the infrastructure to correct the drift.

**A.** False

**B.** True

Explanation

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

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 156**

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

1. max
2. min
3. join
4. replace
5. list
6. length
7. range

**A.** Backends

**B.** Data sources

**C.** Named values

**D.** Functions

**Answer:** ([SHOW ANSWER](#))

Explanation

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not support user-defined functions, and only the functions built into the language are available for use.

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

### **NEW QUESTION: 157**

What is the result of the following terraform function call?

**A.** True

**B.** False

**Answer:** ([SHOW ANSWER](#))

<https://www.terraform.io/docs/configuration/functions/index.html>

### **NEW QUESTION: 158**

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?

**A.** aws\_ami.ubuntu

**B.** data.aws\_ami.ubuntu

C. data.aws\_ami.ubuntu.id

D. aws\_ami.ubuntu.id

**Answer: ([SHOW ANSWER](#))**

Explanation

```
resource "aws_instance" "web" {
```

```
ami= data.aws_ami.ubuntu.id
```

Reference:

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance>

## NEW QUESTION: 159

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. What command will do this?

A. terraform taint

B. terraform apply

C. terraform graph

D. terraform refresh

**Answer: ([SHOW ANSWER](#))**

Explanation

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially new IP address of a tainted server. The plan command will show this if this is the case.

This example will taint a single resource:

```
$ terraform taint aws_security_group.allow_all
```

The resource aws\_security\_group.allow\_all in the module root has been marked as tainted.

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

## NEW QUESTION: 160

A. The ability to update existing infrastructure

B. The ability to delete infrastructure

- C. The ability to reuse best practice configurations and settings
- D. The ability to autoscale a group of servers

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 161**

When should you use the force-unlock command?

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

**Answer:** ([SHOW ANSWER](#))

Manually unlock the state for the defined configuration.

### **NEW QUESTION: 162**

After running into issues with Terraform, you need to enable verbose logging to assist with troubleshooting the error. Which of the following values provides the MOST verbose logging?

- A. ERROR
- B. INFO
- C. WARN
- D. TRACE
- E. DEBUG

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

Examples:

```
export TF_LOG=DEBUG  
export TF_LOG=TRACE
```

### **NEW QUESTION: 163**

Which of the following value will be accepted for var1?

```
variable "var1" {
```

```
  type = string
```

```
}
```

- A. None of the above
- B. Both A and B

C. "5"

D. 5

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform automatically converts number and bool values to strings when needed.

### **NEW QUESTION: 164**

When Terraform needs to be installed in a location where it does not have internet access to download the installer and upgrades, the installation is generally known as to be \_\_\_\_\_.

- A. a private install
- B. disconnected
- C. air-gapped
- D. non-traditional

**Answer:** ([SHOW ANSWER](#))

Explanation

A Terraform Enterprise install that is provisioned on a network that does not have Internet access is generally known as an air-gapped install. These types of installs require you to pull updates, providers, etc. from external sources vs. being able to download them directly.

### **NEW QUESTION: 165**

Which of the following allows Terraform users to apply policy as code to enforce standardized configurations for resources being deployed via infrastructure as code?

- A. Sentinel
- B. Module registry
- C. Functions
- D. Workspaces

**Answer:** ([SHOW ANSWER](#))

Sentinel is a language and framework for policy built to be embedded in existing software to enable fine-grained, logic-based policy decisions. A policy describes under what circumstances certain behaviors are allowed. Sentinel is an enterprise-only feature.

[https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb\\_title](https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb_title)

### **NEW QUESTION: 166**

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

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

#### **NEW QUESTION: 167**

In terraform, most resource dependencies are handled automatically. Which of the following statements describes best how terraform resource dependencies are handled?

- A. Resource dependencies are identified and maintained in a file called `resource.dependencies`. Each terraform provider is required to maintain a list of all resource dependencies for the provider and it's included with the plugin during initialization when terraform init is executed. The file is located in the `terraform.d` folder.
- B. The terraform binary contains a built-in reference map of all defined Terraform resource dependencies. Updates to this dependency map are reflected in terraform versions. To ensure you are working with the latest resource dependency map you must be running the latest version of Terraform.
- C. Resource dependencies are handled automatically by the `depends_on` meta\_argument, which is set to true by default.
- D. 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.

**Answer: D** ([LEAVE A REPLY](#))

<https://www.terraform.io/docs/configuration/resources.html>

#### **NEW QUESTION: 168**

A user runs terraform init on their RHEL based server and per the output, two provider plugins are downloaded: \$ terraform init Initializing the backend...

Initializing provider plugins...

- Checking for available provider plugins...
- Downloading plugin for provider "aws" (hashicorp/aws) 2.44.0...
- Downloading plugin for provider "random" (hashicorp/random) 2.2.1...
- :

Terraform has been successfully initialized! Where are these plugins downloaded to?

- A. The .terraform/plugins directory in the directory terraform init was executed in.
- B. The .terraform.d directory in the directory terraform init was executed in.
- C. The .terraform.plugins directory in the directory terraform init was executed in.
- D. /etc/terraform/plugins

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 169**

What does the command `terraform fmt` do?

- A. Rewrite Terraform configuration files to a canonical format and style.
- B. Deletes the existing configuration file.
- C. Updates the font of the configuration file to the official font supported by HashiCorp.
- D. Formats the state file in order to ensure the latest state of resources can be obtained.

**Answer:** ([SHOW ANSWER](#))

The `terraform fmt` command is used to rewrite Terraform configuration files to a canonical format and style. This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability.

Other Terraform commands that generate Terraform configuration will produce configuration files that conform to the style imposed by `terraform fmt`, so using this style in your own files will ensure consistency.

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

### **NEW QUESTION: 170**

- A. Run `terraform refresh` to update the terraform state , then write the terraform code , and finally run `terraform apply`.
- B. Run `terraform destroy` first since you need to start from fresh every time , before running `terraform apply`.
- C. Write terraform code , and run `terraform push` , to update the terraform state to the remote repo , which in turn will take care of the next steps.
- D. Write the terraform code on the developer machine , run `terraform plan` to check the changes , and run `terraform apply` to provision the infra.

**Answer:** D ([LEAVE A REPLY](#))

You do not need to run `terraform refresh` as `terraform plan` implicitly will run `terraform refresh`.

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

## NEW QUESTION: 171

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. Edit your state file to scrub out the sensitive data
- C. Store the state in an encrypted backend
- D. Always store your secrets in a secrets.tfvars file.

**Answer:** ([SHOW ANSWER](#))

## NEW QUESTION: 172

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. What command will do this?

- A. terraform taint
- B. terraform apply
- C. terraform graph
- D. terraform refresh

**Answer:** ([SHOW ANSWER](#))

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially new IP address of a tainted server. The plan command will show this if this is the case.

This example will taint a single resource:

```
$ terraform taint aws_security_group.allow_all
```

The resource aws\_security\_group.allow\_all in the module root has been marked as tainted.

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

## NEW QUESTION: 173

Any user can publish modules to the public Terraform Module Registry.

- A. True

**B. False**

**Answer: ([SHOW ANSWER](#))**

**NEW QUESTION: 174**

If a DevOps team adopts AWS Cloud Formation as their standardized method for provisioning public cloud resources, which of the following scenarios poses a challenge for this team?

- A.** The organization decides to expand into Azure and wishes to deploy new infrastructure using their existing codebase
- B.** The DevOps team is tasked with automating a manual provisioning process
- C.** The team is asked to manage a new application stack built on AWS-native services
- D.** The team is asked to build a reusable code base that can deploy resources into any AWS region

**Answer: ([SHOW ANSWER](#))**

**NEW QUESTION: 175**

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

**Answer: D ([LEAVE A REPLY](#))**

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>

**NEW QUESTION: 176**

Which of the following commands will launch the Interactive console for Terraform interpolations?

- A.** terraform cli
- B.** terraform cmdline

Explanation

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

- C.** terraform console
- D.** terraform

**Answer: ([SHOW ANSWER](#))**

## **NEW QUESTION: 177**

Module version is required to reference a module on the Terraform Module Registry.

- A. True
- B. False

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (384 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

# HashiCorp.TA-002-P.v2021-05-20.q78

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	78
<b>Version:</b>	v2021-05-20
<b># of views:</b>	2129
<b># of Questions views:</b>	58995

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2021-05-20.q78.html>

## NEW QUESTION: 1

You have been given requirements to create a security group for a new application. Since your organization standardizes on Terraform, you want to add this new security group with the fewest number of lines of code. What feature could you use to iterate over a list of required tcp ports to add to the new security group?

- A. dynamic backend
- B. splat expression
- C. terraform import
- D. dynamic block

**Answer: ([SHOW ANSWER](#))**

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.

<https://www.terraform.io/docs/configuration/expressions.html#dynamic-blocks>

## NEW QUESTION: 2

From the answers below, select the advantages of using Infrastructure as Code.

- A. Provide a codified workflow to develop customer-facing applications.
- B. Safely test modifications using a "dry run" before applying any actual changes.
- C. Easily integrate with application workflows (GitLab Actions, Azure DevOps, CI/CD tools).
- D. Easily change and update existing infrastructure.
- E. Provide reusable modules for easy sharing and collaboration.

**Answer: B,C,D,E ([LEAVE A REPLY](#))**

Infrastructure as Code is not used to develop applications, but it can be used to help deploy or provision those applications to a public cloud provider or on-premises infrastructure.

All of the others are benefits to using Infrastructure as Code over the traditional way of managing infrastructure, regardless if it's public cloud or on-premises.

### **NEW QUESTION: 3**

You have created 2 workspaces PROD and RQA.

You have switched to RQA and provisioned RQA infrastructure from this workspace.

Where is your state file stored?

- A. `terraform.tfstate`
- B. `terraform.tfstate.RQA`
- C. `terraform.d`
- D. `terraform.tfstate.d`

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 4**

Which of the following is considered a Terraform plugin?

- A. Terraform language
- B. Terraform tooling
- C. Terraform logic
- D. Terraform provider

**Answer: D ([LEAVE A REPLY](#))**

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

<https://www.terraform.io/docs/plugins/basics.html>

### **NEW QUESTION: 5**

What resource dependency information is stored in Terraform's state?

- A. Only implicit dependencies are stored in state.
- B. Both implicit and explicit dependencies are stored in state.
- C. Only explicit dependencies are stored in state.
- D. No dependency information is stored in state.

**Answer: ([SHOW ANSWER](#))**

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 6**

What are some of the problems of how infrastructure was traditionally managed before Infrastructure as Code? (select three)

- A. Requests for infrastructure or hardware required a ticket, increasing the time required to deploy applications
- B. Traditional deployment methods are not able to meet the demands of the modern business where resources tend to live days to weeks, rather than months to years
- C. Traditionally managed infrastructure can't keep up with cyclic or elastic applications
- D. Pointing and clicking in a management console is a scalable approach and reduces human error as businesses are moving to a multi-cloud deployment model

**Answer: A,B,C ([LEAVE A REPLY](#))**

Businesses are making a transition where traditionally-managed infrastructure can no longer meet the demands of today's businesses. IT organizations are quickly adopting the public cloud, which is predominantly API-driven. To meet customer demands and save costs, application teams are architecting their applications to support a much higher level of elasticity, supporting technology like containers and public cloud resources. These resources may only live for a matter of hours; therefore the traditional method of raising a ticket to request resources is no longer a viable option. Pointing and clicking in a management console is NOT scale and increases the chance of human error.

### **NEW QUESTION: 7**

You cannot publish your own modules on the Terraform Registry.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 8**

Once a resource is marked as tainted, the next plan will show that the resource will be \_\_\_\_\_ and \_\_\_\_\_ and the next apply will implement this change.

- A. recreated and tainted
- B. destroyed and not recreated
- C. destroyed and recreated
- D. tainted and not destroyed

**Answer: C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 9**

A data block requests that Terraform read from a given data source and export the result under the given local name.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

## **NEW QUESTION: 10**

Which of the following statements best describes the Terraform list(...) type?

- A.** a collection of values where each is identified by a string label.
- B.** a collection of unique values that do not have any secondary identifiers or ordering.
- C.** a sequence of values identified by consecutive whole numbers starting with zero.
- D.** a collection of named attributes that each have their own type.

Explanation

A terraform list is a sequence of values identified by consecutive whole numbers starting with zero. <https://www.terraform.io/docs/configuration/types.html#structural-types>

**Answer: C (LEAVE A REPLY)**

## **NEW QUESTION: 11**

In order to make a Terraform configuration file dynamic and/or reusable, static values should be converted to use what?

- A.** Input Parameters
- B.** Module
- C.** Regular Expressions
- D.** Output Value

**Answer: A (LEAVE A REPLY)**

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module's own source code, and allowing modules to be shared between different configurations.

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

## **NEW QUESTION: 12**

Which of the below options is the equivalent Terraform 0.12 version of the snippet which is written in Terraform 0.11?

"\${var.instance\_id}"

- A.** var.instance\_ids
- B.** variable.instance\_id
- C.** var.instance\_id
- D.** None of the above

**Answer: C (LEAVE A REPLY)**

## **NEW QUESTION: 13**

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.

- A.** False
- B.** True

### **Answer: B ([LEAVE A REPLY](#))**

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.

<https://www.terraform.io/docs/state/sensitive-data.html#recommendations>

### **NEW QUESTION: 14**

State locking does not happen automatically and must be specified at run

- A.** False
- B.** True

### **Answer: A ([LEAVE A REPLY](#))**

State locking happens automatically on all operations that could write state.

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

### **NEW QUESTION: 15**

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a

- 
- A.** First Time Configuration
  - B.** Default Configuration
  - C.** Changing Configuration
  - D.** Partial Configuration
  - E.** Incomplete Configuration

### **Answer: ([SHOW ANSWER](#))**

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process. There are several ways to supply the remaining arguments:

- \* Interactively: Terraform will interactively ask you for the required values, unless interactive input is disabled. Terraform will not prompt for optional values.
- \* 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, in which case it must be downloaded to the local disk before running Terraform.
- \* Command-line key/value pairs: Key/value pairs can be specified via the init command line. Note that many shells retain command-line flags in a history file, so this isn't

recommended for secrets. To specify a single key/value pair, use the -backend-config="KEY=VALUE" option when running terraform init.

<https://www.terraform.io/docs/backends/config.html#partial-configuration>

## NEW QUESTION: 16

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address aws\_instance.web refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.
- C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.
- D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer: A ([LEAVE A REPLY](#))**

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:

module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

aws\_instance.web[3]

Refers to only the last instance in the config, and an address like this:

aws\_instance.web

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

## NEW QUESTION: 17

After creating a new workspace "PROD" you need to run the command terraform select PROD to switch to it.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

By default, when you create a new workspace you are automatically switched to it To create a new workspace and switch to it, you can use terraform workspace new <new\_workspace\_name>; to switch to a existing workspace you can use terraform workspace select <existing\_workspace\_name>; Example:

```
$ terraform workspace new example
```

Created and switched to workspace "example"!

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state for this configuration.

## NEW QUESTION: 18

Your team lead does not trust the junior terraform engineers who now have access to the git repo . So , he wants you to have some sort of a checking layer , whereby , you can ensure that the juniors will not create any non-compliant resources that might lead to a security audit failure in future. What can you do to efficiently enforce this?

- A. Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.
- B. Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them.
- C. Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code.
- D. Create a git master branch , and implement PR . Every change needs to be reviewed by you , before being merged to the master branch.

**Answer: B ([LEAVE A REPLY](#))**

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### NEW QUESTION: 19

Which of the following allows Terraform users to apply policy as code to enforce standardized configurations for resources being deployed via infrastructure as code?

- A. Sentinel
- B. Module registry
- C. Functions
- D. Workspaces

**Answer: A ([LEAVE A REPLY](#))**

Sentinel is a language and framework for policy built to be embedded in existing software to enable fine-grained, logic-based policy decisions. A policy describes under what circumstances certain behaviors are allowed. Sentinel is an enterprise-only feature.

[https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb\\_title](https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb_title)

### NEW QUESTION: 20

Environment variables can be used to set variables. The environment variables must be in the format "\_\_\_\_\_"\_<variablename>. Select the correct prefix string from the following list.

- A. TF\_CLI\_ARGS
- B. TF\_VAR
- C. TF\_VAR\_
- D. TF\_VAR\_ENV

**Answer: C ([LEAVE A REPLY](#))**

Environment variables can be used to set variables. The environment variables must be in the format TF\_VAR\_name and this will be checked last for a value. For example:

```
export TF_VAR_region=us-west-1  
export TF_VAR_ami=ami-049d8641  
export TF_VAR_alist='[1,2,3]'  
export TF_VAR_amap='{ foo = "bar", baz = "qux" }'
```

<https://www.terraform.io/docs/commands/environment-variables.html>

### NEW QUESTION: 21

What kind of resource dependency is stored in terraform.tfstate file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

**Answer: ([SHOW ANSWER](#))**

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a

resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 22**

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer:** ([SHOW ANSWER](#))

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

### **NEW QUESTION: 23**

You have provisioned some aws resources in your test environment through Terraform for a POC work. After the POC, now you want to destroy the resources but before destroying them you want to check what resources will be getting destroyed through terraform. what are the options of doing that? (Select TWO)

- A. Use terraform destroy command
- B. This is not possible
- C. Use terraform plan command
- D. Use terraform plan -destroy command.

**Answer:** ([SHOW ANSWER](#))

<https://learn.hashicorp.com/terraform/getting-started/destroy>

### **NEW QUESTION: 24**

You are reviewing Terraform configurations for a big project in your company. You noticed that there are several identical sets of resources that appear in multiple configurations. What feature of Terraform would you recommend to use to reduce the amount of cloned configuration between the different configurations?

- A. Packages
- B. Backends
- C. Provisioners
- D. Modules

**Answer: D ([LEAVE A REPLY](#))**

Modules are reusable configuration packages that Terraform can share through a variety of sources including Terraform Registries, GitHub, and Amazon S3 buckets.

A module is a container for multiple resources that are used together. Modules can be used to create lightweight abstractions, so that you can describe your infrastructure in terms of its architecture, rather than directly in terms of physical objects.

Modules are reusable configuration packages that Terraform can share through a variety of sources including Terraform Registries, GitHub, and Amazon S3 buckets.

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

**NEW QUESTION: 25**

Your company has been using Terraform Cloud for a some time now . But every team is creating their own modules , and there is no standardization of the modules , with each team creating the resources in their own unique way . You want to enforce a standardization of the modules across the enterprise . What should be your approach.

- A. Create individual workspaces for each team , and ask them to share modules across workspaces.
- B. Implement a Private module registry in Terraform cloud , and ask teams to reference them.
- C. Upgrade to Terraform enterprise , since this is not possible in terraform cloud.
- D. Upload the modules in the terraform public module registry , and ask teams to reference them

**Answer: B ([LEAVE A REPLY](#))**

Terraform Cloud's private module registry helps you share Terraform modules across your organization. It includes support for module versioning, a searchable and filterable list of available modules, and a configuration designer to help you build new workspaces faster. By design, the private module registry works much like the public Terraform Registry. If you're already used the public registry, Terraform Cloud's registry will feel familiar.

Understand the different offerings in Terraform OS, Terraform Cloud and Terraform Enterprise. Terraform Cloud's private module registry helps you share Terraform modules across your organization.

<https://www.terraform.io/docs/cloud/registry/index.html>

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

**NEW QUESTION: 26**

From the code below, identify the implicit dependency:

- A. The S3 bucket labeled company\_data

- B. The EC2 instance labeled web\_server
- C. The EIP with an id of ami-2757f631
- D. The AMI used for the EC2 instance

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 27**

Which of the below backends support state locking?

- A. artifactory
- B. azurerm
- C. consul
- D. S3

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 28**

What is the command you can use to set an environment variable named "var1" of type String?

- A. export TF\_VAR\_VAR1
- B. set TF\_VAR\_var1
- C. variable "var1" { type = "string"}
- D. export TF\_VAR\_var1

**Answer: D ([LEAVE A REPLY](#))**

The environment variable must be in the format TF\_VAR\_name, so for the

### **NEW QUESTION: 29**

Workspaces in Terraform provides similar functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

<https://www.terraform.io/docs/cloud/migrate/workspaces.html>

Workspaces, managed with the `terraform workspace` command, aren't the same thing as Terraform Cloud's workspaces. Terraform Cloud workspaces act more like completely separate working directories; CLI workspaces are just alternate state files.

### **NEW QUESTION: 30**

Jim has created several AWS resources from a single `terraform` configuration file.

Someone from his team has manually modified one of the EC2 instance.

Now to discard the manual change, Jim wants to destroy and recreate the EC2 instance.

What is the best way to do it?

- A. `terraform recreate`
- B. `terraform taint`

C. `terraform destroy`

D. `terraform refresh`

**Answer: B ([LEAVE A REPLY](#))**

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially new IP address of a tainted server. The `plan` command will show this if this is the case.

This example will taint a single resource:

```
$ terraform taint aws_security_group.allow_all
```

The resource `aws_security_group.allow_all` in the module root has been marked as tainted.

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

## NEW QUESTION: 31

A user has created a module called "my\_test\_module" and committed it to GitHub. Over time, several commits have been made with updates to the module, each tagged in GitHub with an incremental version number. Which of the following lines would be required in a module configuration block in terraform to select tagged version v1.0.4?

A. `source = "git::https://example.com/my_test_module.git?ref=v1.0.4"`

Explanation

<https://www.terraform.io/docs/modules/sources.html#selecting-a-revision>

B. `source = "git::https://example.com/my_test_module.git#tag=v1.0.4"`

C. `source = "git::https://example.com/my_test_module.git@tag=v1.0.4"`

D. `source = "git::https://example.com/my_test_module.git&ref=v1.0.4"`

**Answer: A ([LEAVE A REPLY](#))**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

### **NEW QUESTION: 32**

Which of the below are paid features of Terraform Cloud?

- A. Full API Coverage
- B. Secure variable Storage
- C. Roles/ Team management
- D. Cost Estimation
- E. Private Module Registry
- F. Sentinel policies

**Answer: C,D,F ([LEAVE A REPLY](#))**

<https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 33**

In the following code snippet, the block type is identified by which string?

- A. "aws\_instance"
- B. instance\_type
- C. "db"
- D. resource

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 34**

Ric wants to enable detail logging and he wants highest verbosity of logs. Which of the following environment variable settings is correct option for him to select.

- A. Set TF\_LOG = DEBUG
- B. Set VAR\_TF = TRACE
- C. Set TF\_LOG = TRACE
- D. Set VAR\_TF\_LOG = TRACE

**Answer: ([SHOW ANSWER](#))**

<https://www.terraform.io/docs/internals/debugging.html>

### **NEW QUESTION: 35**

True or False? Each Terraform workspace uses its own state file to manage the infrastructure associated with that particular workspace.

- A. False
- B. True

Explanation

The persistent data stored in the backend belongs to a workspace. Initially, the backend has only one workspace, called "default", and thus there is only one Terraform state associated with that configuration.

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 36**

You are using a terraform operation that writes state. Unfortunately automatic state unlocking has failed for that operation. Which of the below commands can be used to remove the already acquired lock on the state?

- A. `terraform unlock`
- B. `terraform force-unlock`
- C. `terraform state unlock`
- D. None of the above

**Answer: B ([LEAVE A REPLY](#))**

Command: `force-unlock`

Manually unlock the state for the defined configuration.

This will not modify your infrastructure. This command removes the lock on the state for the current configuration. The behavior of this lock is dependent on the backend being used. Local state files cannot be unlocked by another process.

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

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

Terraform has a `force-unlock` command to manually unlock the state if unlocking failed. 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.

### **NEW QUESTION: 37**

Terraform Cloud is more powerful when you integrate it with your version control system (VCS) provider. Select all the supported VCS providers from the answers below. (select four)

- A. Bitbucket Cloud
- B. GitHub Enterprise

Explanation

Terraform Cloud supports the following VCS providers:

- <https://www.terraform.io/docs/cloud/vcs/github.html>
- <https://www.terraform.io/docs/cloud/vcs/github.html>
- <https://www.terraform.io/docs/cloud/vcs/github-enterprise.html>
- <https://www.terraform.io/docs/cloud/vcs/gitlab-com.html>
- <https://www.terraform.io/docs/cloud/vcs/gitlab-eece.html>
- <https://www.terraform.io/docs/cloud/vcs/bitbucket-cloud.html>
- <https://www.terraform.io/docs/cloud/vcs/bitbucket-server.html>

- <https://www.terraform.io/docs/cloud/vcs/azure-devops-server.html>
- <https://www.terraform.io/docs/cloud/vcs/azure-devops-services.html>
- <https://www.terraform.io/docs/cloud/vcs/index.html#supported-vcs-providers>

C. Azure DevOps Server

D. GitHub

E. CVS Version Control

**Answer:** A,B,C,D ([LEAVE A REPLY](#))

### **NEW QUESTION: 38**

Which statements best describes what the local variable assignment is doing in the following code snippet:

- A. Create a distinct list of route table name objects
- B. Create a map of route table names to subnet names
- C. Create a map of route table names from a list of subnet names
- D. Create a list of route table names eliminating duplicates

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 39**

Your manager has instructed you to start using terraform for your day-to-day operations, but your security team is concerned about the terraform state files. They have heard it contains confidential information, and are worried that it will not be securely protected.

What should be your response to the security team in this regard?

- A. Inform the security team that using terraform state is optional . There are ways to avoid it , and you will do the same.
- B. Ensure that the state is managed in a remote backend , preferably an enterprise grade state management system like Terraform Cloud.
- C. Mask the confidential entries in the terraform state file , using Vault Enterprise, another Hashicorp product , while keeping it locally.
- D. Keep the state file locally on each developer machine , and ensure that there is a local protection software like KeyPass protecting it.

**Answer:** B ([LEAVE A REPLY](#))

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

State is very important topic for exam. Please read all of the below subtopics Purpose Import Existing Resources Locking Workspaces Remote State Sensitive Data

### **NEW QUESTION: 40**

When using parent/child modules to deploy infrastructure, how would you export a value from one module to import into another module.

For example, a module dynamically deploys an application instance or virtual machine, and you need the IP address in another module to configure a related DNS record in order to reach the newly deployed application.

- A. Export the value using terraform export and input the value using terraform input.
- B. Configure the pertinent provider's configuration with a list of possible IP addresses to use.
- C. Configure an output value in the application module in order to use that value for the DNS module.
- D. Preconfigure the IP address as a parameter in the DNS module.

**Answer:** ([SHOW ANSWER](#))

Output values are like the return values of a Terraform module, and 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`.
- \* When using remote state, root module outputs can be accessed by other configurations via a `terraform_remote_state` data source.

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

## NEW QUESTION: 41

What is the result of the following terraform function call?

- A. True
- B. False

Explanation

<https://www.terraform.io/docs/configuration/functions/index.html>

**Answer:** B ([LEAVE A REPLY](#))

## NEW QUESTION: 42

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

- 1. max
- 2. min
- 3. join
- 4. replace
- 5. list
- 6. length
- 7. range

- A. Backends
- B. Data sources
- C. Named values
- D. Functions

**Answer:** D ([LEAVE A REPLY](#))

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not

support user-defined functions, and only the functions built into the language are available for use.

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

### **NEW QUESTION: 43**

What does terraform plan do ?

- A. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.
- B. Create an execution plan by evaluating the difference between configuration file and state file.
- C. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.
- D. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 44**

Select the most accurate statement to describe the Terraform language from the following list.

- A. Terraform is an immutable, declarative, Infrastructure as Code provisioning language based on Hashicorp Configuration Language, or optionally JSON.
- B. Terraform is a mutable, declarative, Infrastructure as Code configuration management language based on Hashicorp Configuration Language, or optionally JSON.
- C. Terraform is an immutable, procedural, Infrastructure as Code configuration management language based on Hashicorp Configuration Language, or optionally JSON.
- D. Terraform is a mutable, procedural, Infrastructure as Code provisioning language based on Hashicorp Configuration Language, or optionally YAML.

**Answer: A ([LEAVE A REPLY](#))**

Terraform is not a configuration management tool -

<https://www.terraform.io/intro/vs/chefpuppet.html> Terraform is a declarative language -

<https://www.terraform.io/docs/configuration/index.html> Terraform supports a syntax that is JSON compatible - <https://www.terraform.io/docs/configuration/syntax-json.html> Terraform is primarily designed on immutable infrastructure principles -

<https://www.hashicorp.com/resources/what-is-mutable-vs-immutable-infrastructure>

### **NEW QUESTION: 45**

The terraform init 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.

- A. False
- B. True

## **Answer: B ([LEAVE A REPLY](#))**

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

## **NEW QUESTION: 46**

Which of the below options is a valid interpolation syntax for retrieving a data source?

- A. \${google\_storage\_bucket.backend}
- B. \${azurerm\_resource\_group.test.data}
- C. \${aws\_instance.web.id.data}
- D. \${data.google\_dns\_keys.foo\_dns\_keys.key\_signing\_keys[0].ds\_record}

## **Answer: ([SHOW ANSWER](#))**

Data source attributes are interpolated with the general syntax

data.TYPENAME.ATTRIBUTE. The interpolation for a resource is the same but without the data. prefix (TYPENAME.ATTRIBUTE).

<https://www.terraform.io/docs/configuration-0-11/interpolation.html#attributes-of-a-data-source>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## **NEW QUESTION: 47**

1. resource "aws\_s3\_bucket" "example" {
2. bucket = "my-test-s3-terraform-bucket"
3. ...} resource "aws\_iam\_role" "test\_role" {
4. name = "test\_role"
5. ...}

Due to the way that the application code is written, the s3 bucket must be created before the test role is created, otherwise there will be a problem. How can you ensure that?

- A. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- B. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- C. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- D. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

### **Answer: A ([LEAVE A REPLY](#))**

Implicit dependency works only if there is some reference of one resource to another.

Explicit dependency is the option here.

### **NEW QUESTION: 48**

Your team has started using terraform OSS in a big way , and now wants to deploy multi region deployments (DR) in aws using the same terraform files . You want to deploy the same infra (VPC,EC2 ...) in both us-east-1 ,and us-west-2 using the same script , and then peer the VPCs across both the regions to enable DR traffic. But , when you run your script , all resources are getting created in only the default provider region. What should you do? Your provider setting is as below -

# The default provider configuration provider "aws" { region = "us-east-1" }

- A.** No way to enable this via a single script . Write 2 different scripts with different default providers in the 2 scripts , one for us-east , another for us-west.
- B.** Create a list of regions , and then use a for-each to iterate over the regions , and create the same resources ,one after the one , over the loop.
- C.** Use provider alias functionality , and add another provider for us-west region . While creating the resources using the tf script , reference the appropriate provider (using the alias).
- D.** Manually create the DR region , once the Primary has been created , since you are using terraform OSS , and multi region deployment is only available in Terraform Enterprise.

### **Answer: C ([LEAVE A REPLY](#))**

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

# The default provider configuration

```
provider "aws" {
```

```
region = "us-east-1"
```

```
}
```

# Additional provider configuration for west coast region

```
provider "aws" {
```

```
alias = "west"
```

```
region = "us-west-2"
```

```
}
```

<https://www.terraform.io/docs/configuration/providers.html>

## **NEW QUESTION: 49**

A variable az has the following default value. What will be the datatype of the variable?

az=["us-west-1a","us-east-1a"]

- A.** String
- B.** Map
- C.** Object
- D.** List

**Answer:** D ([LEAVE A REPLY](#))

## **NEW QUESTION: 50**

Matt wants to import a manually created EC2 instance into terraform so that he can manage the EC2 instance through terraform going forward. He has written the configuration file of the EC2 instance before importing it to Terraform. Following is the code:

```
resource "aws_instance" "matt_ec2" { ami = "ami-bg2640de" instance_type = "t2.micro"  
vpc_security_group_ids = ["sg-6ae7d613", "sg-53370035"] key_name = "mysecret"  
subnet_id = "subnet-9e3cfbc5" } The instance id of that EC2 instance is  
i-0260835eb7e9bd40 How he can import data of EC2 to state file?
```

- A.** terraform import aws\_instance.id = i-0260835eb7e9bd40
- B.** terraform import i-0260835eb7e9bd40
- C.** terraform import aws\_instance.i-0260835eb7e9bd40
- D.** terraform import aws\_instance.matt\_ec2 i-0260835eb7e9bd40

**Answer:** D ([LEAVE A REPLY](#))

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

## **NEW QUESTION: 51**

What is the result of the following terraform function call?

- A.** goodbye

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

- B.** what?

- C.** hello

**Answer:** ([SHOW ANSWER](#))

## **NEW QUESTION: 52**

Hanah is writing a terraform configuration with nested modules, there are multiple places where she has to use the same conditional expression but she wants to avoid repeating the same values or expressions multiple times in the configuration,. What is a better approach to dealing with this?

- A.** Expressions
- B.** Local Values

**C.** Variables

**D.** Functions

**Answer:** ([SHOW ANSWER](#))

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

### **NEW QUESTION: 53**

Named workspaces are not a suitable isolation mechanism for strong separation between staging and production?

**A.** True

**B.** False

**Answer:** ([SHOW ANSWER](#))

Organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

### **NEW QUESTION: 54**

True or False: A list(...) contain a number of values of the same type while an object(...) can contain a number of values of different types.

**A.** False

**B.** True

**Answer:** **B** ([LEAVE A REPLY](#))

Collection Types

A collection type allows multiple values of one other type to be grouped together as a single value. The type of value within a collection is called its element type. All collection types must have an element type, which is provided as the argument to their constructor. For example, the type `list(string)` means "list of strings", which is a different type than `list(number)`, a list of numbers. All elements of a collection must always be of the same type.

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.

\* set(...): a collection of unique values that do not have any secondary identifiers or ordering.

<https://www.terraform.io/docs/configuration/types.html>

### Structural Types

A structural type allows multiple values of several distinct types to be grouped together as a single value. Structural types require a schema as an argument, to specify which types are allowed for which elements.

The two kinds of structural type in the Terraform language are:

\* object(...): a collection of named attributes that each have their own type.

The schema for object types is { <KEY> = <TYPE>, <KEY> = <TYPE>, ... } - a pair of curly braces containing a comma-separated series of <KEY> = <TYPE> pairs. Values that match the object type must contain all of the specified keys, and the value for each key must match its specified type. (Values with additional keys can still match an object type, but the extra attributes are discarded during type conversion.)

\* tuple(...): a sequence of elements identified by consecutive whole numbers starting with zero, where each element has its own type.

The schema for tuple types is [<TYPE>, <TYPE>, ...] - a pair of square brackets containing a comma-separated series of types. Values that match the tuple type must have exactly the same number of elements (no more and no fewer), and the value in each position must match the specified type for that position.

For example: an object type of object({ name=string, age=number }) would match a value like the following:

```
{  
  name = "John"  
  age = 52  
}
```

Also, an object type of object({ id=string, cidr\_block=string }) would match the object produced by a reference to an aws\_vpc resource, like aws\_vpc.example\_vpc; although the resource has additional attributes, they would be discarded during type conversion.

Finally, a tuple type of tuple([string, number, bool]) would match a value like the following:

```
["a", 15, true]
```

<https://www.terraform.io/docs/configuration/types.html>

## NEW QUESTION: 55

If you enable TF\_LOG = DEBUG, the log will be stored in syslog.log file in the current directory.

A. False

B. True

**Answer: A ([LEAVE A REPLY](#))**

<https://www.terraform.io/docs/internals/debugging.html>

## NEW QUESTION: 56

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A. TF\_TRACE
- B. TF\_DEBUG
- C. TF\_LOG
- D. TF\_INFO

**Answer: C ([LEAVE A REPLY](#))**

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

## NEW QUESTION: 57

When multiple engineers start deploying infrastructure using the same state file, what is a feature of remote state storage that is critical to ensure the state doesn't become corrupt?

- A. Object Storage
- B. State Locking
- C. WorkSpaces
- D. Encryption

**Answer: B ([LEAVE A REPLY](#))**

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.

State locking happens automatically on all operations that could write state. You won't see any message that it is happening. If state locking fails, Terraform will not continue. You can disable state locking for most commands with the -lock flag but it is not recommended.

If acquiring the lock is taking longer than expected, Terraform will output a status message. If Terraform doesn't output a message, state locking is still occurring if your backend supports it.

Not all backends support locking. Please view the list of backend types for details on whether a backend supports locking or not.

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

## NEW QUESTION: 58

Your manager has instructed you to start using terraform for the entire infra provisioning of the application stack. There are 4 environments - DEV , QA , UAT , and PROD. The application team has asked for complete segregation between these environments including the backend , state , and also configurations ,since there will be unique resources

in different environments . What is the possible way to structure the terraform code to facilitate that.

- A.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to a different backend.
- B.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to the same backend.
- C.** Implement terraform workspaces , and map each environment with one workspace.
- D.** Enable remote backend storage . Configure 4 different backend storages , one for each environment.

**Answer:** A ([LEAVE A REPLY](#))

In particular, organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

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

**NEW QUESTION: 59**

What is the default backend for Terraform?

- A.** consul
- B.** gcs
- C.** local
- D.** etcd

**Answer:** ([SHOW ANSWER](#))

By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to.

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

**NEW QUESTION: 60**

How does Terraform handle working with so many providers?

- A.** Terraform ships with all of the plugins embedded in the Terraform binary.
- B.** Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in the configuration's working directory.
- C.** Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in a shared, system-wide plugins directory.
- D.** Terraform allows you to select the providers you want to support during the Terraform installation process.

**Answer:** ([SHOW ANSWER](#))

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

#### **NEW QUESTION: 61**

You wanted to destroy some of the dependent resources from real infrastructure. You choose to delete those resources from your configuration file and run terraform plan and then apply. Which of the following way your resources would be destroyed?

- A.** Terraform can still determine the correct order for destruction from the state even when you delete one or more items from the configuration.
- B.** Those would be destroyed in the order in which they were written in the configuration file previously before you have deleted them from configuration file.
- C.** The resource will be destructed in random order as you have already deleted them from configuration.
- D.** You can not destroy resources by deleting them from configuration file and running plan and apply.

#### **Answer: ([SHOW ANSWER](#))**

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state. Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

#### **NEW QUESTION: 62**

Which of the following state management command allow you to retrieve a list of resources that are part of the state file?

- A.** terraform state list
- B.** terraform state view

C. terraform view

D. terraform list

**Answer: ([SHOW ANSWER](#))**

The terraform state list command is used to list resources within a Terraform state.

Usage: `terraform state list [options] [address...]`

The command will list all resources in the state file matching the given addresses (if any). If no addresses are given, all resources are listed.

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

### **NEW QUESTION: 63**

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A. Explicit use of a provider block in configuration, optionally including a version constraint.
- B. Existence of any resource instance belonging to a particular provider in the current state.
- C. Existence of any provider plugins found locally in the working directory.

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider dependency. The plugin can exist without any reference to it in the terraform configuration. <https://www.terraform.io/docs/commands/providers.html>

D. Use of any resource belonging to a particular provider in a resource or data block in configuration.

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 64**

While using generic git repository as a module source, which of the below options allows terraform to select a specific version or tag instead of selecting the HEAD.

A. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?ref=v1.2.0"}
```

B. Append version argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?version=v1.2.0"}
```

C. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git#ref=v1.2.0"}
```

D. By default, Terraform will clone and use the default branch (referenced by HEAD) in the selected repository and you can not override this.

**Answer: ([SHOW ANSWER](#))**

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:

```
module "vpc" {  
  source = "git::https://example.com/vpc.git?ref=v1.2.0"  
}
```

The value of the ref argument can be any reference that would be accepted by the git checkout command, including branch and tag names.

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

### **NEW QUESTION: 65**

Which of the following actions are performed during a terraform init?

- A. Initializes downloaded and/or installed providers
- B. Initializes the backend configuration
- C. Provisions the declared resources in your configuration
- D. Download the declared providers which are supported by HashiCorp

**Answer: ([SHOW ANSWER](#))**

The terraform init command is used to initialize a working directory containing Terraform configuration files. This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times.

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.

terraform init command does -

- \* Copy a Source Module
- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

### **NEW QUESTION: 66**

Which of the following commands will launch the Interactive console for Terraform interpolations?

- A. `terraform cmdline`

Explanation

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

- B. `terraform`

- C. `terraform console`

- D. `terraform cli`

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 67**

After running into issues with Terraform, you need to enable verbose logging to assist with troubleshooting the error. Which of the following values provides the MOST verbose logging?

- A. `ERROR`

- B. INFO**
- C. WARN**
- D. TRACE**
- E. DEBUG**

**Answer: D ([LEAVE A REPLY](#))**

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

Examples:

```
export TF_LOG=DEBUG  
export TF_LOG=TRACE
```

### **NEW QUESTION: 68**

What are some of the features of Terraform state? (select three)

- A. increased performance**
- B. determining the correct order to destroy resources**
- C. mapping configuration to real-world resources**
- D. inspection of cloud resources**

**Answer: A,C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 69**

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

- A. False**
- B. True**

**Answer: ([SHOW ANSWER](#))**

You can change your backend configuration at any time. You can change both the configuration itself as well as the type of backend (for example from "consul" to "s3"). Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

### **NEW QUESTION: 70**

terraform state subcommands such as list are read-only commands, do read-only commands create state backup files?

- A. Yes**
- B. No**

### **Answer: B ([LEAVE A REPLY](#))**

Subcommands that are read-only (such as list) do not write any backup files since they aren't modifying the state.

All terraform state subcommands that modify the state write backup files. The path of these backup file can be controlled with -backup.

<https://www.terraform.io/docs/commands/state/index.html#backups>

### **NEW QUESTION: 71**

Anyone can publish and share modules on the Terraform Public Module Registry, and meeting the requirements for publishing a module is extremely easy. Select from the following list all valid requirements. (select three)

- A. Module repositories must use this three-part name format, terraform-- .
- B. The registry uses tags to identify module versions.
- C. The module must be on GitHub and must be a public repo.

Explanation

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

- D. The module must be PCI/HIPPA compliant.
- E. Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .

### **Answer: B,C,E ([LEAVE A REPLY](#))**

### **NEW QUESTION: 72**

Which of the following value will be accepted for var1?

```
variable "var1" {  
  type = string  
}
```

- A. None of the above
- B. Both A and B
- C. "5"
- D. 5

### **Answer: B ([LEAVE A REPLY](#))**

Terraform automatically converts number and bool values to strings when needed.

### **NEW QUESTION: 73**

How can you ensure that the engineering team who has access to git repo will not create any non-compliant resources that might lead to a security audit failure in future. your team is using Hashicorp Terraform Enterprise Edition.

- A. Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code.
- B. Implement a review process where every code will be reviewed before merging to the master branch.

**C.** Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them.

**D.** Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.

**Answer:** ([SHOW ANSWER](#))

<https://www.terraform.io/docs/cloud/sentinel/index.html>

## **NEW QUESTION: 74**

lookup retrieves the value of a single element from which of the below data type?

- A.** map
- B.** set
- C.** string
- D.** list

**Answer:** **A** ([LEAVE A REPLY](#))

<https://www.terraform.io/docs/configuration/functions/lookup.html>

## **NEW QUESTION: 75**

What are the benefits of using Infrastructure as Code? (select five)

- A.** Infrastructure as Code is easily repeatable, allowing the user to reuse code to deploy similar, yet different resources
- B.** Infrastructure as Code provides configuration consistency and standardization among deployments
- C.** Infrastructure as Code allows a user to turn a manual task into a simple, automated deployment (Correct) Explanation If you are new to infrastructure as code as a concept, it is the process of managing infrastructure in a file or files rather than manually configuring resources in a user interface.

A resource in this instance is any piece of infrastructure in a given environment, such as a virtual machine, security group, network interface, etc. At a high level, Terraform allows operators to use HCL to author files containing definitions of their desired resources on almost any provider (AWS, GCP, GitHub, Docker, etc) and automates the creation of those resources at the time of application.

- D.** Infrastructure as Code is relatively simple to learn and write, regardless of a user's prior experience with developing code
- E.** Infrastructure as Code gives the user the ability to recreate an application's infrastructure for disaster recovery scenarios
- F.** Infrastructure as Code easily replaces development languages such as Go and .Net for application development

**Answer:** ([SHOW ANSWER](#))

## **NEW QUESTION: 76**

ABC Enterprise has recently tied up with multiple small organizations for exchanging database information. Due to this, the firewall rules are increasing and are more than 100 rules. This is leading firewall configuration file that is difficult to manage. What is the way this type of configuration can be managed easily?

- A. Dynamic Blocks
- B. Terraform Backends
- C. Terraform Functions
- D. Terraform Expression

**Answer: A ([LEAVE A REPLY](#))**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

#### **NEW QUESTION: 77**

Which feature of Terraform allows multiple state files for a single configuration file depending upon the environment?

- A. Terraform Remote Backends
- B. Terraform Workspaces
- C. Terraform Enterprise
- D. Terraform Modules

**Answer: B ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 78**

You want to use terraform import to start managing infrastructure that was not originally provisioned through infrastructure as code. Before you can import the resource's current state, what must you do in order to prepare to manage these resources using Terraform?

- A. Run terraform refresh to ensure that the state file has the latest information for existing resources.
- B. Update the configuration file to include the new resources.
- C. Shut down or stop using the resources being imported so no changes are inadvertently missed.
- D. Modify the Terraform state file to add the new resources.

**Answer: B ([LEAVE A REPLY](#))**

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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. The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import  
successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (**324 Q&As Dumps, 30%OFF**

**Special Discount: freecram)**

# HashiCorp.TA-002-P.v2021-04-20.q98

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	98
<b>Version:</b>	v2021-04-20
<b># of views:</b>	2139
<b># of Questions views:</b>	105115

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2021-04-20.q98.html>

## NEW QUESTION: 1

Complete the following sentence:

For local state, the workspaces are stored directly in a \_\_\_\_\_.

- A. a file called terraform.tfstate.backup
- B. directory called terraform.workspaces.tfstate
- C. a file called terraform.tfstate
- D. directory called terraform.tfstate.d

**Answer:** ([SHOW ANSWER](#))

Explanation

For local state, Terraform stores the workspace states in a directory called terraform.tfstate.d.

<https://www.terraform.io/docs/state/workspaces.html#workspace-internals>

## NEW QUESTION: 2

A user has created three workspaces using the command line - prod, dev, and test. The user wants to create a fourth workspace named stage. Which command will the user execute to accomplish this?

- A. terraform workspace new stage
- B. terraform workspace -new stage
- C. terraform workspace -create stage
- D. terraform workspace create stage

**Answer:** A ([LEAVE A REPLY](#))

Explanation

The terraform workspace new command is used to create a new workspace.

<https://www.terraform.io/docs/commands/workspace/new.html>

## NEW QUESTION: 3

The Terraform language does not support user-defined functions, and so only the functions built in to the language are available for use.

- A. False
- B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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

#### **NEW QUESTION: 4**

When does terraform apply reflect changes in the cloud environment?

- A. None of the above
- B. After updating the state file
- C. However long it takes the resource provider to fulfill the request
- D. Immediately
- E. Based on the value provided to the -refresh command line argument

**Answer: A ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 5**

If you enable TF\_LOG = DEBUG, the log will be stored in syslog.log file in the current directory.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/internals/debugging.html>

#### **NEW QUESTION: 6**

Ric wants to enable detail logging and he wants highest verbosity of logs. Which of the following environment variable settings is correct option for him to select.

- A. Set TF\_LOG = DEBUG
- B. Set VAR\_TF = TRACE
- C. Set TF\_LOG = TRACE
- D. Set VAR\_TF\_LOG = TRACE

**Answer: ([SHOW ANSWER](#))**

Explanation

<https://www.terraform.io/docs/internals/debugging.html>

#### **NEW QUESTION: 7**

What Terraform feature is shown in the example below?

- A. conditional expression
- B. local values

C. dynamic block

D. data source

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 8**

Your team lead does not trust the junior terraform engineers who now have access to the git repo . So , he wants you to have some sort of a checking layer , whereby , you can ensure that the juniors will not create any non-compliant resources that might lead to a security audit failure in future. What can you do to efficiently enforce this?

A. Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.

B. Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them.

C. Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code.

D. Create a git master branch , and implement PR . Every change needs to be reviewed by you , before being merged to the master branch.

**Answer:** B ([LEAVE A REPLY](#))

Explanation

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### **NEW QUESTION: 9**

Command terraform refresh will update state file?

A. False

B. True

**Answer:** B ([LEAVE A REPLY](#))

Explanation

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. If the state is changed, this may cause changes to occur during the next plan or apply.

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

### **NEW QUESTION: 10**

After running into issues with Terraform, you need to enable verbose logging to assist with troubleshooting the error. Which of the following values provides the MOST verbose logging?

- A. ERROR
- B. INFO
- C. WARN
- D. TRACE
- E. DEBUG

**Answer: D ([LEAVE A REPLY](#))**

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

Examples:

```
export TF_LOG=DEBUG  
export TF_LOG=TRACE
```

### **NEW QUESTION: 11**

Terraform works well in Windows but a Windows server is required.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

You may see this question in actual exam. Please remember : Terraform does not require GO language to be installed as a prerequisite and it does not require a Windows Server as well.

### **NEW QUESTION: 12**

terraform state subcommands such as list are read-only commands, do read-only commands create state backup files?

- A. Yes
- B. No

**Answer: ([SHOW ANSWER](#))**

Explanation

Subcommands that are read-only (such as list) do not write any backup files since they aren't modifying the state.

All terraform state subcommands that modify the state write backup files. The path of these backup file can be controlled with -backup.

<https://www.terraform.io/docs/commands/state/index.html#backups>

### **NEW QUESTION: 13**

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

**Answer: ([SHOW ANSWER](#))**

Explanation

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>

### **NEW QUESTION: 14**

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

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 15**

By default, where does Terraform store its state file?

- A. Amazon S3 bucket
- B. shared directory
- C. remotely using Terraform Cloud
- D. current working directory

**Answer: ([SHOW ANSWER](#))**

Explanation

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

### **NEW QUESTION: 16**

A single terraform resource file that defines an aws\_instance resource can simply be renamed to vsphere\_virtual\_machine in order to switch cloud providers.

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Every provider has its own required and allowed declarations none of which match between cloud providers.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF** **Special Discount: freecram**)

### NEW QUESTION: 17

Which of the following represents a feature of Terraform Cloud that is NOT free to customers?

- A. Roles and Team Management
- B. WorkSpace Management
- C. Private Module Registry
- D. VCS Integration

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Role Based Access Controls (RBAC) for controlling permissions for who has access to what configurations within an organization and it is not free to customers.

<https://www.hashicorp.com/products/terraform/pricing/>

### NEW QUESTION: 18

Your organization has moved to AWS and has manually deployed infrastructure using the console. Recently, a decision has been made to standardize on Terraform for all deployments moving forward.

What can you do to ensure that all existing is managed by Terraform moving forward without interruption to existing services?

- A. Submit a ticket to AWS and ask them to export the state of all existing resources and use terraform import to import them into the state file.
- B. Delete the existing resources and recreate them using new a Terraform configuration so Terraform can manage them moving forward.
- C. Resources that are manually deployed in the AWS console cannot be imported by Terraform.
- D. Using terraform import, import the existing infrastructure into your Terraform state.

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

## NEW QUESTION: 19

Matt wants to import a manually created EC2 instance into terraform so that he can manage the EC2 instance through terraform going forward. He has written the configuration file of the EC2 instance before importing it to Terraform. Following is the code:

```
resource "aws_instance" "matt_ec2" { ami = "ami-bg2640de" instance_type = "t2.micro"  
vpc_security_group_ids = ["sg-6ae7d613", "sg-53370035"] key_name = "mysecret"  
subnet_id =  
"subnet-9e3cfbc5" }
```

The instance id of that EC2 instance is i-0260835eb7e9bd40 How he can import data of EC2 to state file?

- A. `terraform import aws_instance.id = i-0260835eb7e9bd40`
- B. `terraform import i-0260835eb7e9bd40`
- C. `terraform import aws_instance.i-0260835eb7e9bd40`
- D. `terraform import aws_instance.matt_ec2 i-0260835eb7e9bd40`

**Answer: D ([LEAVE A REPLY](#))**

Explanation

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

## NEW QUESTION: 20

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

**Answer: D ([LEAVE A REPLY](#))**

## NEW QUESTION: 21

Which of the following Terraform files should be ignored by Git when committing code to a repo? (select Three)

- A. Files named exactly `terraform.tfvars` or `terraform.tfvars.json`.
- B. Any files with names ending in `.auto.tfvars` or `.auto.tfvars.json`.
- C. `input.tf`
- D. `terraform.tfstate`
- E. `output.tf`

**Answer: A,B,D ([LEAVE A REPLY](#))**

Explanation

The `.gitignore` file should be configured to ignore Terraform files that either contain sensitive data or are not required to save.

Terraform state (`terraform.tfstate`) can contain sensitive data, depending on the resources in use and your definition of "sensitive." The state contains resource IDs and all resource attributes. For resources such as databases, this may contain initial passwords.

When using local state, state is stored in plain-text JSON files.

The `terraform.tfvars` file may contain sensitive data, such as passwords or IP addresses of an environment that you may not want to share with others.

## NEW QUESTION: 22

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

**Answer:** A ([LEAVE A REPLY](#))

### **NEW QUESTION: 23**

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 = var.num\_servers
- B.** servers = variable.num\_servers
- C.** servers = num\_servers
- D.** servers = var(num\_servers)

**Answer:** C ([LEAVE A REPLY](#))

### **NEW QUESTION: 24**

In Terraform Enterprise, a workspace can be mapped to how many VCS repos?

- A.** 5
- B.** 2
- C.** 3
- D.** 1

**Answer:** D ([LEAVE A REPLY](#))

Explanation

A workspace can only be configured to a single VCS repo, however, multiple workspaces can use the same repo.

<https://www.terraform.io/docs/cloud/workspaces/vcs.html>

### **NEW QUESTION: 25**

State is a requirement for Terraform to function

- A.** True
- B.** False

**Answer:** ([SHOW ANSWER](#))

Explanation

State is a necessary requirement for Terraform to function. It is often asked if it is possible for Terraform to work without state, or for Terraform to not use state and just inspect cloud resources on every run.

Purpose of Terraform State

State is a necessary requirement for Terraform to function. It is often asked if it is possible for Terraform to work without state, or for Terraform to not use state and just inspect cloud resources on every run. This page will help explain why Terraform state is required.

As you'll see from the reasons below, state is required. And in the scenarios where Terraform may be able to get away without state, doing so would require shifting massive amounts of complexity from one place (state) to another place (the replacement concept).

## 1. Mapping to the Real World

Terraform requires some sort of database to map Terraform config to the real world. When you have a resource resource "aws\_instance" "foo" in your configuration, Terraform uses this map to know that instance i- abcd1234 is represented by that resource.

For some providers like AWS, Terraform could theoretically use something like AWS tags. Early prototypes of Terraform actually had no state files and used this method. However, we quickly ran into problems. The first major issue was a simple one: not all resources support tags, and not all cloud providers support tags.

Therefore, for mapping configuration to resources in the real world, Terraform uses its own state structure.

## 2. Metadata

Alongside the mappings between resources and remote objects, Terraform must also track metadata such as resource dependencies.

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

One way to avoid this would be for Terraform to know a required ordering between resource types. For example, Terraform could know that servers must be deleted before the subnets they are a part of. The complexity for this approach quickly explodes, however: in addition to Terraform having to understand the ordering semantics of every resource for every cloud, Terraform must also understand the ordering across providers.

Terraform also stores other metadata for similar reasons, such as a pointer to the provider configuration that was most recently used with the resource in situations where multiple aliased providers are present.

## 3. Performance

In addition to basic mapping, Terraform stores a cache of the attribute values for all resources in the state. This is the most optional feature of Terraform state and is done only as a performance improvement.

When running a terraform plan, Terraform must know the current state of resources in order to effectively determine the changes that it needs to make to reach your desired configuration.

For small infrastructures, Terraform can query your providers and sync the latest attributes from all your resources. This is the default behavior of Terraform: for every plan and apply, Terraform will sync all resources in your state.

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to query multiple resources at once, and the round trip time for each resource is hundreds of milliseconds. On top of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

#### 4. Syncing

In the default configuration, Terraform stores the state in a file in the current working directory where Terraform was run. This is okay for getting started, but when using Terraform in a team it is important for everyone to be working with the same state so that operations will be applied to the same remote objects.

Remote state is the recommended solution to this problem. With a fully-featured state backend, Terraform can use remote locking as a measure to avoid two or more different users accidentally running Terraform at the same time, and thus ensure that each Terraform run begins with the most recent updated state.

### NEW QUESTION: 26

Valarie has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code. Valarie wants to hide the output value in the CLI after terraform apply that's why she has used sensitive parameter.

```
1. output "db_password" {  
2.   value = local.db_password  
3.   sensitive = true  
4. }
```

Since sensitive is set to true, will the value associated with db password be available in plain-text in the state file for everyone to read?

- A. Yes
- B. No

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Outputs can be marked as containing sensitive material by setting the `sensitive` attribute to true, like this:

```
output "sensitive" {  
  sensitive = true  
  value = VALUE  
}
```

When outputs are displayed on-screen following a `terraform apply` or `terraform refresh`, sensitive outputs are redacted, with `<sensitive>` displayed in place of their value.

Limitations of Sensitive Outputs

The values of sensitive outputs are still stored in the Terraform state, and available using the `terraform output` command, so cannot be relied on as a sole means of protecting values.

Sensitivity is not tracked internally, so if the output is interpolated in another module into a resource, the value will be displayed.

### **NEW QUESTION: 27**

Select two answers to complete the following sentence: Before a new provider can be used, it must be \_\_\_\_\_ and \_\_\_\_\_.

- A. approved by HashiCorp
- B. uploaded to source control
- C. declared in the configuration
- D. initialized

**Answer:** ([SHOW ANSWER](#))

Explanation

Each time a new provider is added to configuration -- either explicitly via a provider block or by adding a resource from that provider -- Terraform must initialize the provider before it can be used. Initialization downloads and installs the provider's plugin so that it can later be executed.

### **NEW QUESTION: 28**

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

**Answer:** E ([LEAVE A REPLY](#))

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

### **NEW QUESTION: 29**

A "backend" in Terraform determines how state is loaded and how an operation such as `apply` is executed.

Which of the following is not a supported backend type?

- A. Terraform enterprise
- B. Consul
- C. Github
- D. S3
- E. Artifactory

**Answer:** C ([LEAVE A REPLY](#))

## Explanation

Github is not a supported backend type.

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

## NEW QUESTION: 30

What is a downside to using the Vault provider to read secrets from Vault?

- A. Secrets are persisted to the state file and plans.
- B. Terraform and Vault must be running on the same version.
- C. Terraform and Vault must be running on the same physical host.
- D. Terraform requires a unique auth method to work with Vault.

**Answer: A ([LEAVE A REPLY](#))**

## Explanation

The Vault provider allows Terraform to read from, write to, and configure Hashicorp Vault. Interacting with Vault from Terraform causes any secrets that you read and write to be persisted in both Terraform's state file and in any generated plan files. For any Terraform module that reads or writes Vault secrets, these files should be treated as sensitive and protected accordingly.

## NEW QUESTION: 31

What is the provider for this fictitious resource?

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

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

**Answer: C ([LEAVE A REPLY](#))**

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**

## **NEW QUESTION: 32**

Your manager has instructed you to start using terraform for the entire infra provisioning of the application stack. There are 4 environments - DEV , QA , UAT , and PROD. The application team has asked for complete segregation between these environments including the backend , state , and also configurations ,since there will be unique resources in different environments . What is the possible way to structure the terraform code to facilitate that.

- A.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to a different backend.
- B.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to the same backend.
- C.** Implement terraform workspaces , and map each environment with one workspace.
- D.** Enable remote backend storage . Configure 4 different backend storages , one for each environment.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

In particular, organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

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

## **NEW QUESTION: 33**

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

- A.** True
- B.** False

**Answer: B ([LEAVE A REPLY](#))**

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

## **NEW QUESTION: 34**

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A.** Explicit use of a provider block in configuration, optionally including a version constraint.
- B.** Use of any resource belonging to a particular provider in a resource or data block in configuration.
- C.** Existence of any resource instance belonging to a particular provider in the current state.

D. Existence of any provider plugins found locally in the working directory.

**Answer: A,B,C ([LEAVE A REPLY](#))**

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider dependency. The plugin can exist without any reference to it in the terraform configuration.

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

### **NEW QUESTION: 35**

What are some of the problems of how infrastructure was traditionally managed before Infrastructure as Code?

(select three)

- A. Requests for infrastructure or hardware required a ticket, increasing the time required to deploy applications
- B. Traditional deployment methods are not able to meet the demands of the modern business where resources tend to live days to weeks, rather than months to years
- C. Traditionally managed infrastructure can't keep up with cyclic or elastic applications
- D. Pointing and clicking in a management console is a scalable approach and reduces human error as businesses are moving to a multi-cloud deployment model

**Answer: A,B,C ([LEAVE A REPLY](#))**

Explanation

Businesses are making a transition where traditionally-managed infrastructure can no longer meet the demands of today's businesses. IT organizations are quickly adopting the public cloud, which is predominantly API-driven. To meet customer demands and save costs, application teams are architecting their applications to support a much higher level of elasticity, supporting technology like containers and public cloud resources.

These resources may only live for a matter of hours; therefore the traditional method of raising a ticket to request resources is no longer a viable option. Pointing and clicking in a management console is NOT scale and increases the chance of human error.

### **NEW QUESTION: 36**

Consider the following Terraform 0.12 configuration snippet:

```
1. variable "vpc_cidrs" {  
2.   type = map  
3.   default = {  
4.     us-east-1 = "10.0.0.0/16"  
5.     us-east-2 = "10.1.0.0/16"  
6.     us-west-1 = "10.2.0.0/16"  
7.     us-west-2 = "10.3.0.0/16"  
8.   }  
9. }
```

10.  
11. resource "aws\_vpc" "shared" {  
12. cidr\_block = \_\_\_\_\_  
13. }

How would you define the cidr\_block for us-east-1 in the aws\_vpc resource using a variable?

- A. var.vpc\_cidrs[0]
- B. var.vpc\_cidrs["us-east-1"]
- C. var.vpc\_cidrs.0
- D. vpc\_cidrs["us-east-1"]

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 37**

Which of the following challenges would Terraform be a candidate for solving? (Select THREE)

- A. Have a single interoperable tool to manage the variety of services including GitHub repositories, MySQL database, and Kubernetes clusters.
- B. Reduce the number of workflows needed for managing infrastructure across each of the companies public and private clouds.
- C. Enable self-service infrastructure to allocate resources on your proprietary private cloud.
- D. Utilize a single tool for all of the infrastructure and configuration management needs.

**Answer: A,B,C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 38**

lookup retrieves the value of a single element from which of the below data type?

- A. map
- B. set
- C. string
- D. list

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

### **NEW QUESTION: 39**

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

- 1. max
- 2. min
- 3. join
- 4. replace
- 5. list

6. length
  7. range
- A. Backends
  - B. Data sources
  - C. Named values
  - D. Functions

**Answer: D ([LEAVE A REPLY](#))**

Explanation

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not support user-defined functions, and only the functions built into the language are available for use.

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

#### **NEW QUESTION: 40**

Which of the following are string functions? Select three

- A. tostring
- B. tonumber
- C. Chomp
- D. format
- E. join

**Answer: C,D,E ([LEAVE A REPLY](#))**

Explanation

tonumber and tostring are Type Conversion function

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

#### **NEW QUESTION: 41**

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

**Answer: B ([LEAVE A REPLY](#))**

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

#### **NEW QUESTION: 42**

When should you use the force-unlock command?

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

D. Your apply failed due to a state lock

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Manually unlock the state for the defined configuration.

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

### **NEW QUESTION: 43**

A user creates three workspaces from the command line - prod, dev, and test. Which of the following commands will the user run to switch to the dev workspace?

- A. terraform workspace dev
- B. terraform workspace select dev
- C. terraform workspace -switch dev
- D. terraform workspace switch dev

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The terraform workspace select command is used to choose a different workspace to use for further operations.

<https://www.terraform.io/docs/commands/workspace/select.html>

### **NEW QUESTION: 44**

Your company has been using Terraform Cloud for a some time now . But every team is creating their own modules , and there is no standardization of the modules , with each team creating the resources in their own unique way . You want to enforce a standardization of the modules across the enterprise . What should be your approach.

- A. Create individual workspaces for each team , and ask them to share modules across workspaces.
- B. Implement a Private module registry in Terraform cloud , and ask teams to reference them.
- C. Upgrade to Terraform enterprise , since this is not possible in terraform cloud.
- D. Upload the modules in the terraform public module registry , and ask teams to reference them

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Terraform Cloud's private module registry helps you share Terraform modules across your organization. It includes support for module versioning, a searchable and filterable list of available modules, and a configuration designer to help you build new workspaces faster. By design, the private module registry works much like the public Terraform Registry. If you're already used the public registry, Terraform Cloud's registry will feel familiar.

Understand the different offerings in Terraform OS, Terraform Cloud and Terraform Enterprise. Terraform Cloud's private module registry helps you share Terraform modules across your organization.

<https://www.terraform.io/docs/cloud/registry/index.html>

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

### NEW QUESTION: 45

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. Terraform attempts to provision the resource up to three times before exiting with an error
- B. the terraform plan is rolled back and all provisioned resources are removed
- C. it is automatically deleted
- D. the resource is marked as tainted

**Answer:** D ([LEAVE A REPLY](#))

Explanation

If a resource successfully creates but fails during provisioning, Terraform will error and mark the resource as

"tainted". A resource that is tainted has been physically created, but can't be considered safe to use since provisioning failed. Terraform also does not automatically roll back and destroy the resource during the apply when the failure happens, because that would go against the execution plan: the execution plan would've said a resource will be created, but does not say it will ever be deleted.

### NEW QUESTION: 46

Which of the following is considered a Terraform plugin?

- A. Terraform language
- B. Terraform tooling
- C. Terraform logic
- D. Terraform provider

**Answer:** ([SHOW ANSWER](#))

Explanation

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

<https://www.terraform.io/docs/plugins/basics.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

**TA-002-P exam questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

### **NEW QUESTION: 47**

Only the user that generated a plan may apply it.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The optional -out argument can be used to save the generated plan to a file for later execution with terraform apply, which can be useful when running Terraform in automation.

Reference: <https://learn.hashicorp.com/tutorials/terraform/automate-terraform>

### **NEW QUESTION: 48**

Which of the following statements best describes the Terraform list(...) type?

- A. a collection of values where each is identified by a string label.
- B. a sequence of values identified by consecutive whole numbers starting with zero.
- C. a collection of unique values that do not have any secondary identifiers or ordering.
- D. a collection of named attributes that each have their own type.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

A terraform list is a sequence of values identified by consecutive whole numbers starting with zero.

<https://www.terraform.io/docs/configuration/types.html#structural-types>

### **NEW QUESTION: 49**

What kind of resource dependency is stored in terraform.tfstate file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 50**

In the example below, where is the value of the DNS record's IP address originating from?

```
1. resource "aws_route53_record" "www"
2. {
3.   zone_id = aws_route53_zone.primary.zone_id
4.   name = "www.example.com"
5.   type = "A"
6.   ttl = "300"
7.   records = [module.web_server.instance_ip_address]
8. }
```

- A. The regular expression named module.web\_server
- B. The output of a module named web\_server
- C. By querying the AWS EC2 API to retrieve the IP address
- D. Value of the web\_server parameter from the variables.tf file

**Answer:** ([SHOW ANSWER](#))

Explanation

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\_address, you could access that value as module.web\_server.instance\_ip\_address.

## NEW QUESTION: 51

HashiCorp offers multiple versions of Terraform, including Terraform open-source, Terraform Cloud, and Terraform Enterprise. Which of the following Terraform features are only available in the Enterprise edition?

(select four)

- A. SAML/SSO
- B. Sentinel
- C. Audit Logs
- D. Clustering
- E. Private Module Registry
- F. Private Network Connectivity

**Answer:** A,C,F ([LEAVE A REPLY](#))

Explanation

While there are a ton of features that are available to open source users, many features that are part of the Enterprise offering are geared towards larger teams and enterprise functionality. To see what specific features are part of Terraform Cloud and Terraform Enterprise, check out this link.

<https://www.hashicorp.com/products/terraform/pricing/>

## NEW QUESTION: 52

True or False? Each Terraform workspace uses its own state file to manage the infrastructure associated with that particular workspace.

- A. False
- B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The persistent data stored in the backend belongs to a workspace. Initially, the backend has only one workspace, called "default", and thus there is only one Terraform state associated with that configuration.

### **NEW QUESTION: 53**

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

**Answer: C ([LEAVE A REPLY](#))**

Explanation

terraform workspace list

The command will list all existing workspaces.

Reference: <https://www.terraform.io/docs/cli/commands/workspace/list.html>

### **NEW QUESTION: 54**

Select all features which are exclusive to Terraform Enterprise. (Select Three)

- A. Sentinel
- B. Cost Estimation
- C. Audit Logs
- D. Clustering
- E. SAML/SSO

**Answer: C,D,E ([LEAVE A REPLY](#))**

Explanation

Sentinel and Cost Estimation are also available in Terraform Cloud

<https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 55**

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

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

**Answer: D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 56**

What does the command terraform fmt do?

- A. Rewrite Terraform configuration files to a canonical format and style.
- B. Deletes the existing configuration file.
- C. Updates the font of the configuration file to the official font supported by HashiCorp.
- D. Formats the state file in order to ensure the latest state of resources can be obtained.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The terraform fmt command is used to rewrite Terraform configuration files to a canonical format and style.

This command applies a subset of the Terraform language style conventions, along with other minor adjustments for readability.

Other Terraform commands that generate Terraform configuration will produce configuration files that conform to the style imposed by terraform fmt, so using this style in your own files will ensure consistency.

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

### **NEW QUESTION: 57**

Terraform providers are always installed from the Internet.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

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>

### **NEW QUESTION: 58**

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

## NEW QUESTION: 59

When configuring a remote backend in Terraform, it might be a good idea to purposely omit some of the required arguments to ensure secrets and other important data aren't inadvertently shared with others. What are the ways the remaining configuration can be added to Terraform so it can initialize and communicate with the backend? (select three)

- A. directly querying HashiCorp Vault for the secrets
- B. command-line key/value pairs
- C. use the `-backend-config=PATH` to specify a separate config file
- D. interactively on the command line

**Answer: B,C,D ([LEAVE A REPLY](#))**

Explanation

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process. There are several ways to supply the remaining arguments:

<https://www.terraform.io/docs/backends/init.html#backend-initialization>

## NEW QUESTION: 60

Where does the Terraform local backend store its state?

- A. In the `/tmp` directory
- B. In the `terraform.tfvars` file
- C. In the `terraform.tfstate` file
- D. In the user's `.terraformrc` file

**Answer: ([SHOW ANSWER](#))**

Explanation

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>

## NEW QUESTION: 61

What does terraform refresh command do?

- A. terraform refresh syncs the state file with the real world infrastructure.
- B. terraform refresh command basically updates the configuration file with the current state of the actual infrastructure
- C. terraform refresh is used to change/modify the infrastructure based on the existing state file, at that moment.
- D. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- E. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.

**Answer:** ([SHOW ANSWER](#))

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

## NEW QUESTION: 62

True or False? By default, Terraform destroy will prompt for confirmation before proceeding.

- A. False

- B. True

**Answer:** B ([LEAVE A REPLY](#))

## NEW QUESTION: 63

Once a resource is marked as tainted, the next plan will show that the resource will be \_\_\_\_\_ and \_\_\_\_\_

\_\_\_\_\_ and the next apply will implement this change.

- A. tainted and not destroyed
- B. destroyed and recreated
- C. destroyed and not recreated
- D. recreated and tainted

**Answer:** B ([LEAVE A REPLY](#))

## NEW QUESTION: 64

Refer to the following terraform variable definition

```
variable "track_tag" { type = list default = ["data_ec2","integration_ec2","digital_ec2"]}  
track_tag = { Name = element(var.track_tag,count.index)} If count.index is set to 2, which  
of the following values will be assigned to the name attribute of track_tag variable?
```

- A. data\_ec2
- B. track\_tag
- C. integration\_ec2
- D. digital\_ec2

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 65**

You want to get involved in the development of Terraform. As this is an open source project, you would like to contribute a fix for an open issue of Terraform. What programming language will need to use to write the fix?

- A. It depends on which command issue related to.
- B. Python
- C. Go
- D. Java

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Basic programming knowledge. Terraform and Terraform Plugins are written in the Go programming language, but even if you've never written a line of Go before, you're still welcome to take a dive into the code and submit patches. The community is happy to assist with code reviews and offer guidance specific to Go.

### **NEW QUESTION: 66**

Talal is a DevOps engineer and he has deployed the production infrastructure using Terraform. He is using a very large configuration file to maintain and update the actual infrastructure. As the infrastructure have grown to a very complex and large, he has started experiencing slowness when he runs terraform plan. What are the options for him to resolve this slowness?

- A. Use -refresh=true flag as well as the -target flag with terraform plan in order to work around this.
- B. Run terraform refresh every time before running terraform plan.
- C. Break large configurations into several smaller configurations that can each be independently applied.
- D. Use -refresh=false flag as well as the -target flag with terraform plan in order to work around this.

**Answer: C,D ([LEAVE A REPLY](#))**

Explanation

For larger infrastructures, querying every resource is too slow. Many cloud providers do not provide APIs to query multiple resources at once, and the round trip time for each

resource is hundreds of milliseconds. On top of this, cloud providers almost always have API rate limiting so Terraform can only request a certain number of resources in a period of time. Larger users of Terraform make heavy use of the `-refresh=false` flag as well as the `-target` flag in order to work around this. In these scenarios, the cached state is treated as the record of truth.

Although 'Use `-refresh=false` flag as well as the `-target` flag with terraform plan in order to work around this.' is a solution, but its not always recommended. Instead of using `-target` as a means to operate on isolated portions of very large configurations, prefer instead to break large configurations into several smaller configurations that can each be independently applied. Data sources can be used to access information about resources created in other configurations, allowing a complex system architecture to be broken down into more manageable parts that can be updated independently.

Option 'Run terraform refresh every time before running terraform plan.' and 'Use `-refresh=true` flag as well as the `-target` flag with terraform plan in order to work around this.' is not correct because in both the cases terraform will query every resources of the infrastructure.

### **NEW QUESTION: 67**

What is not processed when running a terraform refresh?

- A. State file
- B. Configuration file
- C. Credentials
- D. Cloud provider

**Answer: C,D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 68**

By default, provisioners that fail will also cause the Terraform apply itself to error. How can you change this default behavior within a provisioner?

- A. provisioner "local-exec" { on\_failure = "next" }
- B. provisioner "local-exec" { when = "failure" terraform apply }
- C. provisioner "local-exec" { on\_failure = "continue" }
- D. provisioner "local-exec" { on\_failure = continue }

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 69**

What does terraform plan do ?

- A. Create an execution plan by evaluating the difference between configuration file and state file.

- B. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.
- C. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.
- D. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.

**Answer:** A ([LEAVE A REPLY](#))

#### **NEW QUESTION: 70**

Which of the following Terraform commands will automatically refresh the state unless supplied with additional flags or arguments? Choose TWO correct answers.

- A. `terraform state`
- B. `terraform output`
- C. `terraform validate`
- D. `terraform plan`
- E. `terraform apply`

**Answer:** ([SHOW ANSWER](#))

#### **NEW QUESTION: 71**

You have created two workspaces PROD and DEV. You have switched to DEV and provisioned DEV infrastructure from this workspace. Where is your state file stored?

- A. `terraform.d`
- B. `terraform.tfstate`
- C. `terraform.tfstate.DEV`
- D. `terraform.tfstate.d`

**Answer:** D ([LEAVE A REPLY](#))

Explanation

Terraform stores the workspace states in a directory called `terraform.tfstate.d`. This directory should be treated similarly to default workspace state file `terraform.tfstate` `main.tf` provider.`tf` `terraform.tfstate.d` DEV `terraform.tfstate` # DEV workspace state file PROD `terraform.tfstate` # PROD workspace state file `terraform.tfvars` # Default workspace state file `variables.tf`

#### **NEW QUESTION: 72**

The following is a snippet from a Terraform configuration file:

Which, when validated, results in the following error:

Fill in the blank in the error message with the correct string from the list below.

- A. `version`
- B. `multi`
- C. `label`
- D. `alias`

**Answer: D ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-providerinstances>

### **NEW QUESTION: 73**

Why might a user opt to include the following snippet in their configuration file?

- A. Terraform 0.12 introduced substantial changes to the syntax used to write Terraform configuration
- B. The user wants to ensure that the application being deployed is a minimum version of 0.12
- C. this ensures that all Terraform providers are above a certain version to match the application being deployed
- D. versions before Terraform 0.12 were not approved by HashiCorp to be used in production

**Answer: A ([LEAVE A REPLY](#))**

### **NEW QUESTION: 74**

What is the default backend for Terraform?

- A. consul
- B. gcs
- C. local
- D. etcd

**Answer: ([SHOW ANSWER](#))**

Explanation

By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to.

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

### **NEW QUESTION: 75**

Which task does terraform init not perform?

- 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

**Answer: D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 76**

When Terraform needs to be installed in a location where it does not have internet access to download the installer and upgrades, the installation is generally known as to be \_\_\_\_\_.

- A. a private install
- B. disconnected
- C. air-gapped
- D. non-traditional

**Answer:** ([SHOW ANSWER](#))

Explanation

A Terraform Enterprise install that is provisioned on a network that does not have Internet access is generally known as an air-gapped install. These types of installs require you to pull updates, providers, etc. from external sources vs. being able to download them directly.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 77

Which of the below features of Terraform can be used for managing small differences between different environments which can act more like completely separate working directories.

- A. Repositories
- B. Workspaces
- C. Environment Variables
- D. Backends

**Answer:** B ([LEAVE A REPLY](#))

Explanation

workspaces allow conveniently switching between multiple instances of a single configuration within its single backend. They are convenient in a number of situations, but cannot solve all problems.

A common use for multiple workspaces is to create a parallel, distinct copy of a set of infrastructure in order to test a set of changes before modifying the main production infrastructure. For example, a developer working on a complex set of infrastructure changes might create a new temporary workspace in order to freely experiment with changes without affecting the default workspace.

Non-default workspaces are often related to feature branches in version control. The default workspace might correspond to the "master" or "trunk" branch, which describes the intended state of production infrastructure.

When a feature branch is created to develop a change, the developer of that feature might create a corresponding workspace and deploy into it a temporary "copy" of the main infrastructure so that changes can be tested without affecting the production infrastructure. Once the change is merged and deployed to the default workspace, the test infrastructure can be destroyed and the temporary workspace deleted.

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

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

## NEW QUESTION: 78

Which of the following best describes the default local backend?

- A. The local backend is where Terraform Enterprise stores logs to be processed by an log collector.
- B. The local backend stores state on the local filesystem, locks the state using system APIs, and performs operations locally.
- C. The local backend is the directory where resources deployed by Terraform have direct access to in order to update their current state.
- D. The local backend is how Terraform connects to public cloud services, such as AWS, Azure, or GCP.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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

```
terraform {  
  backend "local" {  
    path = "relative/path/to/terraform.tfstate"  
  }  
}
```

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

## NEW QUESTION: 79

Your team has started using terraform OSS in a big way , and now wants to deploy multi region deployments (DR) in aws using the same terraform files . You want to deploy the same infra (VPC,EC2 ...) in both us-east-1 ,and us-west-2 using the same script , and then peer the VPCs across both the regions to enable DR traffic. But , when you run your script , all resources are getting created in only the default provider region.

What should you do? Your provider setting is as below -

```
# The default provider configuration provider "aws" { region = "us-east-1" }
```

- A.** No way to enable this via a single script . Write 2 different scripts with different default providers in the  
2 scripts , one for us-east , another for us-west.
- B.** Create a list of regions , and then use a for-each to iterate over the regions , and create the same resources ,one after the one , over the loop.
- C.** Use provider alias functionality , and add another provider for us-west region . While creating the resources using the tf script , reference the appropriate provider (using the alias).
- D.** Manually create the DR region , once the Primary has been created , since you are using terraform OSS , and multi region deployment is only available in Terraform Enterprise.

**Answer: C ([LEAVE A REPLY](#))**

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
alias = "west"
region = "us-west-2"
}
```

<https://www.terraform.io/docs/configuration/providers.html>

## NEW QUESTION: 80

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A.** TF\_TRACE
- B.** TF\_DEBUG
- C.** TF\_LOG
- D.** TF\_INFO

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

## NEW QUESTION: 81

Select the operating systems which are supported for a clustered Terraform Enterprise:

(select four)

- A. Unix
- B. Red Hat
- C. CentOS
- D. Amazon Linux
- E. Ubuntu

**Answer: B,C,D,E ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/enterprise/before-installing/index.html#operating-systemrequirements>

## NEW QUESTION: 82

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address aws\_instance.web refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.
- C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.
- D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer: ([SHOW ANSWER](#))**

Explanation

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:  
module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

`aws_instance.web[3]`

Refers to only the last instance in the config, and an address like this:

`aws_instance.web`

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

### **NEW QUESTION: 83**

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

**Answer: A,B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 84**

Which of the following variable definition files will terraform load automatically?

- A. `terraform.tfvar`
- B. Any files with names ending in `.auto.tfvars.json`
- C. `terraform.tfvars`
- D. `terraform.tfvars.json`

**Answer: B,C,D ([LEAVE A REPLY](#))**

Explanation

Terraform also automatically loads a number of variable definitions files if they are present:

Files named exactly `terraform.tfvars` or `terraform.tfvars.json`.

Any files with names ending in `.auto.tfvars` or `.auto.tfvars.json`.

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

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

### **NEW QUESTION: 85**

What is the standard workflow that a developer follows while working with terraform open source version?

- A. Run terraform refresh to update the terraform state , then write the terraform code , and finally run terraform apply.
- B. Run terraform destroy first since you need to start from fresh every time , before running terraform apply.
- C. Write terraform code , and run terraform push , to update the terraform state to the remote repo , which in turn will take care of the next steps.
- D. Write the terraform code on the developer machine , run terraform plan to check the changes , and run terraform apply to provision the infra.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

You do not need to run terraform refresh as terraform plan implicitly will run terraform refresh.

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

### **NEW QUESTION: 86**

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

**Answer: B,D ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 87**

Which feature of Terraform allows multiple state files for a single configuration file depending upon the environment?

- A. Terraform Enterprise
- B. Terraform Remote Backends
- C. Terraform Modules
- D. Terraform Workspaces

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 88**

You have created a custom variable definition file testing.tfvars. How will you use it for provisioning infrastructure?

- A. `terraform apply -var-state-file ="testing.tfvars"`
- B. `terraform plan -var-file="testing.tfvar"`

- C. `terraform apply -var-file="testing.tfvars"`
- D. `terraform apply var-file="testing.tfvars"`

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 89**

Which of the below configuration file formats are supported by Terraform? (Select TWO)

- A. Node
- B. JSON
- C. Go
- D. YAML
- E. HCL

**Answer: B,E ([LEAVE A REPLY](#))**

Explanation

Terraform supports both HashiCorp Configuration Language (HCL) and JSON formats for configurations.

<https://www.terraform.io/docs/configuration/>

### **NEW QUESTION: 90**

You have configured an Auto Scaling group in AWS to automatically scale the number of instances behind a load balancer based on the instances CPU utilization. The instances are configured using a Launch Configuration. You have observed that the Auto Scaling group doesn't successfully scale when you apply changes that require replacing the Launch Configuration. Why is this happening?

- A. You need to configure an explicit dependency for the Auto Scaling group using the `depends_on` meta-parameter.
- B. You need to configure an explicit dependency for the Launch Configuration using the `depends_on` meta-parameter.
- C. You need to configure the Auto Scaling group's `create_before_destroy` meta-parameter.
- D. You need to configure the Launch Configuration's `create_before_destroy` meta-parameter.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

[https://www.terraform.io/docs/providers/aws/r/launch\\_configuration.html#using-withautoscaling-groups](https://www.terraform.io/docs/providers/aws/r/launch_configuration.html#using-withautoscaling-groups)

### **NEW QUESTION: 91**

Terraform Enterprise (also referred to as pTFE) requires what type of backend database for a clustered deployment?

- A. PostgreSQL

- B. Cassandra
- C. MySQL
- D. MSSQL

**Answer:** ([SHOW ANSWER](#))

Explanation

External Services mode stores the majority of the stateful data used by the instance in an external PostgreSQL database and an external S3-compatible endpoint or Azure blob storage. There is still critical data stored on the instance that must be managed with snapshots. Be sure to check the PostgreSQL Requirements for information that needs to be present for Terraform Enterprise to work. This option is best for users with expertise managing PostgreSQL or users that have access to managed PostgreSQL offerings like AWS RDS.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

## NEW QUESTION: 92

While Terraform is generally written using the HashiCorp Configuration Language (HCL), what other syntax can Terraform be expressed in?

- A. JSON
- B. YAML
- C. TypeScript
- D. XML

**Answer:** A ([LEAVE A REPLY](#))

Explanation

The constructs in the Terraform language can also be expressed in JSON syntax, which is harder for humans to read and edit but easier to generate and parse programmatically.

## NEW QUESTION: 93

You have created an AWS EC2 instance of type t2.micro through your terraform configuration file ec2.tf .

Now you want to change the instance type from t2.micro to t2.medium. Accordingly you have changed your configuration file and ran terraform plan. After running terraform plan you check the output and saw one instance will be updated from t2.micro --> t2.medium. After this you went to grab a coffee without running terraform apply and

meanwhile a member of your team changed the instance type of that EC2 instance to t2.medium from aws console. After coming to your desk you run terraform apply. What will happen?

- A. 1 resource will be updated and you will see the message : Apply Complete !  
Resources : 0 added, 1 changed, 0 destroyed.
- B. No resource will be updated and you will see the message : Apply Complete !  
Resources : 0 added, 0 changed, 0 destroyed.
- C. terraform apply will through an error.
- D. The instance type will be changed to t2.micro and again will be changed to t2.medium

**Answer: B ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 94**

A variable az has the following default value. What will be the datatype of the variable?

az=["us-west-1a","us-east-1a"]

- A. Object
- B. String
- C. List
- D. Map

**Answer: C ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 95**

ABC Enterprise has recently tied up with multiple small organizations for exchanging database information.

Due to this, the firewall rules are increasing and are more than 100 rules. This is leading firewall configuration file that is difficult to manage. What is the way this type of configuration can be managed easily?

- A. Terraform Functions
- B. Terraform Backends
- C. Dynamic Blocks
- D. Terraform Expression

**Answer: ([SHOW ANSWER](#))**

#### **NEW QUESTION: 96**

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 plan 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.

**Answer: C ([LEAVE A REPLY](#))**

Reference:

<https://www.google.com/search?q=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new+infrastructure.&oq=Write+a+Terraform+configuration%2C+run+terraform+init%2C+run+terraform+plan+to+view+planned+infrastructure+changes%2C+and+terraform+apply+to+create+new+infrastructure.&aqs=chrome..69i57.556j0j7&sourceid=chrome&ie=UTF-8>

## NEW QUESTION: 97

When writing Terraform code, HashiCorp recommends that you use how many spaces between each nesting level?

- A. 0
- B. 1
- C. 2
- D. 4

**Answer: ([SHOW ANSWER](#))**

Explanation

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.

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:

```
ami = "abc123"  
instance_type = "t2.micro"
```

When both arguments and blocks appear together inside a block body, place all of the arguments together at the top and then place nested blocks below them. Use one blank line to separate the arguments from the blocks.

Use empty lines to separate logical groups of arguments within a block.

For blocks that contain both arguments and "meta-arguments" (as defined by the Terraform language semantics), list meta-arguments first and separate them from other arguments with one blank line. Place meta-argument blocks last and separate them from other blocks with one blank line.

```
resource "aws_instance" "example" {
```

```
count = 2 # meta-argument first
ami = "abc123"
instance_type = "t2.micro"
network_interface {
# ...
}
lifecycle { # meta-argument block last
create_before_destroy = true
}
```

Top-level blocks should always be separated from one another by one blank line. Nested blocks should also be separated by blank lines, except when grouping together related blocks of the same type (like multiple provisioner blocks in a resource).

Avoid separating multiple blocks of the same type with other blocks of a different type, unless the block types are defined by semantics to form a family. (For example: root\_block\_device, ebs\_block\_device and ephemeral\_block\_device on aws\_instance form a family of block types describing AWS block devices, and can therefore be grouped together and mixed.)

## NEW QUESTION: 98

In terraform, most resource dependencies are handled automatically. Which of the following statements describes best how terraform resource dependencies are handled?

- A.** Resource dependencies are identified and maintained in a file called resource.dependencies. Each terraform provider is required to maintain a list of all resource dependencies for the provider and it's included with the plugin during initialization when terraform init is executed. The file is located in the terraform.d folder.
- B.** The terraform binary contains a built-in reference map of all defined Terraform resource dependencies.
- C.** Resource dependencies are handled automatically by the depends\_on meta\_argument, which is set to true by default.
- D.** 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.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/resources.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!  
Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com  
**TA-002-P exam questions have been updated and answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

# HashiCorp.TA-002-P.v2021-05-20.q78

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	78
<b>Version:</b>	v2021-05-20
<b># of views:</b>	2129
<b># of Questions views:</b>	58995

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2021-05-20.q78.html>

## NEW QUESTION: 1

You have been given requirements to create a security group for a new application. Since your organization standardizes on Terraform, you want to add this new security group with the fewest number of lines of code. What feature could you use to iterate over a list of required tcp ports to add to the new security group?

- A. dynamic backend
- B. splat expression
- C. terraform import
- D. dynamic block

**Answer: ([SHOW ANSWER](#))**

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.

<https://www.terraform.io/docs/configuration/expressions.html#dynamic-blocks>

## NEW QUESTION: 2

From the answers below, select the advantages of using Infrastructure as Code.

- A. Provide a codified workflow to develop customer-facing applications.
- B. Safely test modifications using a "dry run" before applying any actual changes.
- C. Easily integrate with application workflows (GitLab Actions, Azure DevOps, CI/CD tools).
- D. Easily change and update existing infrastructure.
- E. Provide reusable modules for easy sharing and collaboration.

**Answer: B,C,D,E ([LEAVE A REPLY](#))**

Infrastructure as Code is not used to develop applications, but it can be used to help deploy or provision those applications to a public cloud provider or on-premises infrastructure.

All of the others are benefits to using Infrastructure as Code over the traditional way of managing infrastructure, regardless if it's public cloud or on-premises.

### **NEW QUESTION: 3**

You have created 2 workspaces PROD and RQA.

You have switched to RQA and provisioned RQA infrastructure from this workspace.

Where is your state file stored?

- A. `terraform.tfstate`
- B. `terraform.tfstate.RQA`
- C. `terraform.d`
- D. `terraform.tfstate.d`

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 4**

Which of the following is considered a Terraform plugin?

- A. Terraform language
- B. Terraform tooling
- C. Terraform logic
- D. Terraform provider

**Answer: D ([LEAVE A REPLY](#))**

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

<https://www.terraform.io/docs/plugins/basics.html>

### **NEW QUESTION: 5**

What resource dependency information is stored in Terraform's state?

- A. Only implicit dependencies are stored in state.
- B. Both implicit and explicit dependencies are stored in state.
- C. Only explicit dependencies are stored in state.
- D. No dependency information is stored in state.

**Answer: ([SHOW ANSWER](#))**

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 6**

What are some of the problems of how infrastructure was traditionally managed before Infrastructure as Code? (select three)

- A. Requests for infrastructure or hardware required a ticket, increasing the time required to deploy applications
- B. Traditional deployment methods are not able to meet the demands of the modern business where resources tend to live days to weeks, rather than months to years
- C. Traditionally managed infrastructure can't keep up with cyclic or elastic applications
- D. Pointing and clicking in a management console is a scalable approach and reduces human error as businesses are moving to a multi-cloud deployment model

**Answer: A,B,C ([LEAVE A REPLY](#))**

Businesses are making a transition where traditionally-managed infrastructure can no longer meet the demands of today's businesses. IT organizations are quickly adopting the public cloud, which is predominantly API-driven. To meet customer demands and save costs, application teams are architecting their applications to support a much higher level of elasticity, supporting technology like containers and public cloud resources. These resources may only live for a matter of hours; therefore the traditional method of raising a ticket to request resources is no longer a viable option. Pointing and clicking in a management console is NOT scale and increases the chance of human error.

### **NEW QUESTION: 7**

You cannot publish your own modules on the Terraform Registry.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

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

### **NEW QUESTION: 8**

Once a resource is marked as tainted, the next plan will show that the resource will be \_\_\_\_\_ and \_\_\_\_\_ and the next apply will implement this change.

- A. recreated and tainted
- B. destroyed and not recreated
- C. destroyed and recreated
- D. tainted and not destroyed

**Answer: C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 9**

A data block requests that Terraform read from a given data source and export the result under the given local name.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

## **NEW QUESTION: 10**

Which of the following statements best describes the Terraform list(...) type?

- A.** a collection of values where each is identified by a string label.
- B.** a collection of unique values that do not have any secondary identifiers or ordering.
- C.** a sequence of values identified by consecutive whole numbers starting with zero.
- D.** a collection of named attributes that each have their own type.

Explanation

A terraform list is a sequence of values identified by consecutive whole numbers starting with zero. <https://www.terraform.io/docs/configuration/types.html#structural-types>

**Answer: C (LEAVE A REPLY)**

## **NEW QUESTION: 11**

In order to make a Terraform configuration file dynamic and/or reusable, static values should be converted to use what?

- A.** Input Parameters
- B.** Module
- C.** Regular Expressions
- D.** Output Value

**Answer: A (LEAVE A REPLY)**

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module's own source code, and allowing modules to be shared between different configurations.

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

## **NEW QUESTION: 12**

Which of the below options is the equivalent Terraform 0.12 version of the snippet which is written in Terraform 0.11?

"\${var.instance\_id}"

- A.** var.instance\_ids
- B.** variable.instance\_id
- C.** var.instance\_id
- D.** None of the above

**Answer: C (LEAVE A REPLY)**

## **NEW QUESTION: 13**

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.

- A.** False
- B.** True

**Answer: B ([LEAVE A REPLY](#))**

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.

<https://www.terraform.io/docs/state/sensitive-data.html#recommendations>

**NEW QUESTION: 14**

State locking does not happen automatically and must be specified at run

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

State locking happens automatically on all operations that could write state.

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

**NEW QUESTION: 15**

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a

- 
- A. First Time Configuration
  - B. Default Configuration
  - C. Changing Configuration
  - D. Partial Configuration
  - E. Incomplete Configuration

**Answer: ([SHOW ANSWER](#))**

You do not need to specify every required argument in the backend configuration. Omitting certain arguments may be desirable to avoid storing secrets, such as access keys, within the main configuration. When some or all of the arguments are omitted, we call this a partial configuration.

With a partial configuration, the remaining configuration arguments must be provided as part of the initialization process. There are several ways to supply the remaining arguments:

- \* Interactively: Terraform will interactively ask you for the required values, unless interactive input is disabled. Terraform will not prompt for optional values.
- \* 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, in which case it must be downloaded to the local disk before running Terraform.
- \* Command-line key/value pairs: Key/value pairs can be specified via the init command line. Note that many shells retain command-line flags in a history file, so this isn't

recommended for secrets. To specify a single key/value pair, use the -backend-config="KEY=VALUE" option when running terraform init.

<https://www.terraform.io/docs/backends/config.html#partial-configuration>

## NEW QUESTION: 16

Given the below resource configuration -

```
resource "aws_instance" "web" { # ... count = 4 }
```

What does the terraform resource address aws\_instance.web refer to?

- A. It refers to all 4 web instances , together , for further individual segregation , indexing is required , with a 0 based index.
- B. It refers to the last web EC2 instance , as by default , if no index is provided , the last / N-1 index is used.
- C. It refers to the first web EC2 instance out of the 4 ,as by default , if no index is provided , the first / 0th index is used.
- D. The above will result in a syntax error , as it is not syntactically correct . Resources defined using count , can only be referenced using indexes.

**Answer: A ([LEAVE A REPLY](#))**

A Resource Address is a string that references a specific resource in a larger infrastructure. An address is made up of two parts:

[module path][resource spec]

Module path:

A module path addresses a module within the tree of modules. It takes the form:

module.A.module.B.module.C...

Multiple modules in a path indicate nesting. If a module path is specified without a resource spec, the address applies to every resource within the module. If the module path is omitted, this addresses the root module.

Given a Terraform config that includes:

```
resource "aws_instance" "web" {  
# ...  
count = 4  
}
```

An address like this:

aws\_instance.web[3]

Refers to only the last instance in the config, and an address like this:

aws\_instance.web

Refers to all four "web" instances.

<https://www.terraform.io/docs/internals/resource-addressing.html>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

## NEW QUESTION: 17

After creating a new workspace "PROD" you need to run the command terraform select PROD to switch to it.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

By default, when you create a new workspace you are automatically switched to it To create a new workspace and switch to it, you can use terraform workspace new <new\_workspace\_name>; to switch to a existing workspace you can use terraform workspace select <existing\_workspace\_name>; Example:

```
$ terraform workspace new example
```

Created and switched to workspace "example"!

You're now on a new, empty workspace. Workspaces isolate their state, so if you run "terraform plan" Terraform will not see any existing state for this configuration.

## NEW QUESTION: 18

Your team lead does not trust the junior terraform engineers who now have access to the git repo . So , he wants you to have some sort of a checking layer , whereby , you can ensure that the juniors will not create any non-compliant resources that might lead to a security audit failure in future. What can you do to efficiently enforce this?

- A. Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.
- B. Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them.
- C. Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code.
- D. Create a git master branch , and implement PR . Every change needs to be reviewed by you , before being merged to the master branch.

**Answer: B ([LEAVE A REPLY](#))**

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### NEW QUESTION: 19

Which of the following allows Terraform users to apply policy as code to enforce standardized configurations for resources being deployed via infrastructure as code?

- A. Sentinel
- B. Module registry
- C. Functions
- D. Workspaces

**Answer: A ([LEAVE A REPLY](#))**

Sentinel is a language and framework for policy built to be embedded in existing software to enable fine-grained, logic-based policy decisions. A policy describes under what circumstances certain behaviors are allowed. Sentinel is an enterprise-only feature.

[https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb\\_title](https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb_title)

### NEW QUESTION: 20

Environment variables can be used to set variables. The environment variables must be in the format "\_\_\_\_\_"\_<variablename>. Select the correct prefix string from the following list.

- A. TF\_CLI\_ARGS
- B. TF\_VAR
- C. TF\_VAR\_
- D. TF\_VAR\_ENV

**Answer: C ([LEAVE A REPLY](#))**

Environment variables can be used to set variables. The environment variables must be in the format TF\_VAR\_name and this will be checked last for a value. For example:

```
export TF_VAR_region=us-west-1  
export TF_VAR_ami=ami-049d8641  
export TF_VAR_alist='[1,2,3]'  
export TF_VAR_amap='{ foo = "bar", baz = "qux" }'
```

<https://www.terraform.io/docs/commands/environment-variables.html>

### NEW QUESTION: 21

What kind of resource dependency is stored in terraform.tfstate file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

**Answer: ([SHOW ANSWER](#))**

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a

resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 22**

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer:** ([SHOW ANSWER](#))

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

### **NEW QUESTION: 23**

You have provisioned some aws resources in your test environment through Terraform for a POC work. After the POC, now you want to destroy the resources but before destroying them you want to check what resources will be getting destroyed through terraform. what are the options of doing that? (Select TWO)

- A. Use terraform destroy command
- B. This is not possible
- C. Use terraform plan command
- D. Use terraform plan -destroy command.

**Answer:** ([SHOW ANSWER](#))

<https://learn.hashicorp.com/terraform/getting-started/destroy>

### **NEW QUESTION: 24**

You are reviewing Terraform configurations for a big project in your company. You noticed that there are several identical sets of resources that appear in multiple configurations. What feature of Terraform would you recommend to use to reduce the amount of cloned configuration between the different configurations?

- A. Packages
- B. Backends
- C. Provisioners
- D. Modules

**Answer: D ([LEAVE A REPLY](#))**

Modules are reusable configuration packages that Terraform can share through a variety of sources including Terraform Registries, GitHub, and Amazon S3 buckets.

A module is a container for multiple resources that are used together. Modules can be used to create lightweight abstractions, so that you can describe your infrastructure in terms of its architecture, rather than directly in terms of physical objects.

Modules are reusable configuration packages that Terraform can share through a variety of sources including Terraform Registries, GitHub, and Amazon S3 buckets.

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

**NEW QUESTION: 25**

Your company has been using Terraform Cloud for a some time now . But every team is creating their own modules , and there is no standardization of the modules , with each team creating the resources in their own unique way . You want to enforce a standardization of the modules across the enterprise . What should be your approach.

- A. Create individual workspaces for each team , and ask them to share modules across workspaces.
- B. Implement a Private module registry in Terraform cloud , and ask teams to reference them.
- C. Upgrade to Terraform enterprise , since this is not possible in terraform cloud.
- D. Upload the modules in the terraform public module registry , and ask teams to reference them

**Answer: B ([LEAVE A REPLY](#))**

Terraform Cloud's private module registry helps you share Terraform modules across your organization. It includes support for module versioning, a searchable and filterable list of available modules, and a configuration designer to help you build new workspaces faster. By design, the private module registry works much like the public Terraform Registry. If you're already used the public registry, Terraform Cloud's registry will feel familiar.

Understand the different offerings in Terraform OS, Terraform Cloud and Terraform Enterprise. Terraform Cloud's private module registry helps you share Terraform modules across your organization.

<https://www.terraform.io/docs/cloud/registry/index.html>

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

**NEW QUESTION: 26**

From the code below, identify the implicit dependency:

- A. The S3 bucket labeled company\_data

- B. The EC2 instance labeled web\_server
- C. The EIP with an id of ami-2757f631
- D. The AMI used for the EC2 instance

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 27**

Which of the below backends support state locking?

- A. artifactory
- B. azurerm
- C. consul
- D. S3

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 28**

What is the command you can use to set an environment variable named "var1" of type String?

- A. export TF\_VAR\_VAR1
- B. set TF\_VAR\_var1
- C. variable "var1" { type = "string"}
- D. export TF\_VAR\_var1

**Answer: D ([LEAVE A REPLY](#))**

The environment variable must be in the format TF\_VAR\_name, so for the

### **NEW QUESTION: 29**

Workspaces in Terraform provides similar functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.

- A. True
- B. False

**Answer: B ([LEAVE A REPLY](#))**

<https://www.terraform.io/docs/cloud/migrate/workspaces.html>

Workspaces, managed with the `terraform workspace` command, aren't the same thing as Terraform Cloud's workspaces. Terraform Cloud workspaces act more like completely separate working directories; CLI workspaces are just alternate state files.

### **NEW QUESTION: 30**

Jim has created several AWS resources from a single `terraform` configuration file.

Someone from his team has manually modified one of the EC2 instance.

Now to discard the manual change, Jim wants to destroy and recreate the EC2 instance.

What is the best way to do it?

- A. `terraform recreate`
- B. `terraform taint`

C. `terraform destroy`

D. `terraform refresh`

**Answer: B ([LEAVE A REPLY](#))**

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

This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted. Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement this change.

Forcing the recreation of a resource is useful when you want a certain side effect of recreation that is not visible in the attributes of a resource. For example: re-running provisioners will cause the node to be different or rebooting the machine from a base image will cause new startup scripts to run.

Note that tainting a resource for recreation may affect resources that depend on the newly tainted resource. For example, a DNS resource that uses the IP address of a server may need to be modified to reflect the potentially new IP address of a tainted server. The `plan` command will show this if this is the case.

This example will taint a single resource:

```
$ terraform taint aws_security_group.allow_all
```

The resource `aws_security_group.allow_all` in the module root has been marked as tainted.

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

## NEW QUESTION: 31

A user has created a module called "my\_test\_module" and committed it to GitHub. Over time, several commits have been made with updates to the module, each tagged in GitHub with an incremental version number. Which of the following lines would be required in a module configuration block in terraform to select tagged version v1.0.4?

A. `source = "git::https://example.com/my_test_module.git?ref=v1.0.4"`

Explanation

<https://www.terraform.io/docs/modules/sources.html#selecting-a-revision>

B. `source = "git::https://example.com/my_test_module.git#tag=v1.0.4"`

C. `source = "git::https://example.com/my_test_module.git@tag=v1.0.4"`

D. `source = "git::https://example.com/my_test_module.git&ref=v1.0.4"`

**Answer: A ([LEAVE A REPLY](#))**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!  
Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com  
**TA-002-P exam questions have been updated and answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram**)

### **NEW QUESTION: 32**

Which of the below are paid features of Terraform Cloud?

- A. Full API Coverage
- B. Secure variable Storage
- C. Roles/ Team management
- D. Cost Estimation
- E. Private Module Registry
- F. Sentinel policies

**Answer: C,D,F ([LEAVE A REPLY](#))**

<https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 33**

In the following code snippet, the block type is identified by which string?

- A. "aws\_instance"
- B. instance\_type
- C. "db"
- D. resource

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 34**

Ric wants to enable detail logging and he wants highest verbosity of logs. Which of the following environment variable settings is correct option for him to select.

- A. Set TF\_LOG = DEBUG
- B. Set VAR\_TF = TRACE
- C. Set TF\_LOG = TRACE
- D. Set VAR\_TF\_LOG = TRACE

**Answer: ([SHOW ANSWER](#))**

<https://www.terraform.io/docs/internals/debugging.html>

### **NEW QUESTION: 35**

True or False? Each Terraform workspace uses its own state file to manage the infrastructure associated with that particular workspace.

- A. False
- B. True

Explanation

The persistent data stored in the backend belongs to a workspace. Initially, the backend has only one workspace, called "default", and thus there is only one Terraform state associated with that configuration.

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 36**

You are using a terraform operation that writes state. Unfortunately automatic state unlocking has failed for that operation. Which of the below commands can be used to remove the already acquired lock on the state?

- A. `terraform unlock`
- B. `terraform force-unlock`
- C. `terraform state unlock`
- D. None of the above

**Answer: B ([LEAVE A REPLY](#))**

Command: `force-unlock`

Manually unlock the state for the defined configuration.

This will not modify your infrastructure. This command removes the lock on the state for the current configuration. The behavior of this lock is dependent on the backend being used. Local state files cannot be unlocked by another process.

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

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

Terraform has a `force-unlock` command to manually unlock the state if unlocking failed. 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.

### **NEW QUESTION: 37**

Terraform Cloud is more powerful when you integrate it with your version control system (VCS) provider. Select all the supported VCS providers from the answers below. (select four)

- A. Bitbucket Cloud
- B. GitHub Enterprise

Explanation

Terraform Cloud supports the following VCS providers:

- <https://www.terraform.io/docs/cloud/vcs/github.html>
- <https://www.terraform.io/docs/cloud/vcs/github.html>
- <https://www.terraform.io/docs/cloud/vcs/github-enterprise.html>
- <https://www.terraform.io/docs/cloud/vcs/gitlab-com.html>
- <https://www.terraform.io/docs/cloud/vcs/gitlab-eece.html>
- <https://www.terraform.io/docs/cloud/vcs/bitbucket-cloud.html>
- <https://www.terraform.io/docs/cloud/vcs/bitbucket-server.html>

- <https://www.terraform.io/docs/cloud/vcs/azure-devops-server.html>
- <https://www.terraform.io/docs/cloud/vcs/azure-devops-services.html>
- <https://www.terraform.io/docs/cloud/vcs/index.html#supported-vcs-providers>

**C.** Azure DevOps Server

**D.** GitHub

**E.** CVS Version Control

**Answer:** A,B,C,D ([LEAVE A REPLY](#))

### **NEW QUESTION: 38**

Which statements best describes what the local variable assignment is doing in the following code snippet:

- A.** Create a distinct list of route table name objects
- B.** Create a map of route table names to subnet names
- C.** Create a map of route table names from a list of subnet names
- D.** Create a list of route table names eliminating duplicates

**Answer:** ([SHOW ANSWER](#))

### **NEW QUESTION: 39**

Your manager has instructed you to start using terraform for your day-to-day operations, but your security team is concerned about the terraform state files. They have heard it contains confidential information, and are worried that it will not be securely protected.

What should be your response to the security team in this regard?

- A.** Inform the security team that using terraform state is optional . There are ways to avoid it , and you will do the same.
- B.** Ensure that the state is managed in a remote backend , preferably an enterprise grade state management system like Terraform Cloud.
- C.** Mask the confidential entries in the terraform state file , using Vault Enterprise, another Hashicorp product , while keeping it locally.
- D.** Keep the state file locally on each developer machine , and ensure that there is a local protection software like KeyPass protecting it.

**Answer:** B ([LEAVE A REPLY](#))

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

State is very important topic for exam. Please read all of the below subtopics Purpose Import Existing Resources Locking Workspaces Remote State Sensitive Data

### **NEW QUESTION: 40**

When using parent/child modules to deploy infrastructure, how would you export a value from one module to import into another module.

For example, a module dynamically deploys an application instance or virtual machine, and you need the IP address in another module to configure a related DNS record in order to reach the newly deployed application.

- A. Export the value using terraform export and input the value using terraform input.
- B. Configure the pertinent provider's configuration with a list of possible IP addresses to use.
- C. Configure an output value in the application module in order to use that value for the DNS module.
- D. Preconfigure the IP address as a parameter in the DNS module.

**Answer:** ([SHOW ANSWER](#))

Output values are like the return values of a Terraform module, and 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`.
- \* When using remote state, root module outputs can be accessed by other configurations via a `terraform_remote_state` data source.

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

## NEW QUESTION: 41

What is the result of the following terraform function call?

- A. True
- B. False

Explanation

<https://www.terraform.io/docs/configuration/functions/index.html>

**Answer:** B ([LEAVE A REPLY](#))

## NEW QUESTION: 42

Select the feature below that best completes the sentence:

The following list represents the different types of \_\_\_\_\_ available in Terraform.

- 1. max
- 2. min
- 3. join
- 4. replace
- 5. list
- 6. length
- 7. range

- A. Backends
- B. Data sources
- C. Named values
- D. Functions

**Answer:** D ([LEAVE A REPLY](#))

The Terraform language includes a number of built-in functions that you can call from within expressions to transform and combine values. The Terraform language does not

support user-defined functions, and only the functions built into the language are available for use.

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

### **NEW QUESTION: 43**

What does terraform plan do ?

- A. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.
- B. Create an execution plan by evaluating the difference between configuration file and state file.
- C. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.
- D. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.

**Answer: B ([LEAVE A REPLY](#))**

### **NEW QUESTION: 44**

Select the most accurate statement to describe the Terraform language from the following list.

- A. Terraform is an immutable, declarative, Infrastructure as Code provisioning language based on Hashicorp Configuration Language, or optionally JSON.
- B. Terraform is a mutable, declarative, Infrastructure as Code configuration management language based on Hashicorp Configuration Language, or optionally JSON.
- C. Terraform is an immutable, procedural, Infrastructure as Code configuration management language based on Hashicorp Configuration Language, or optionally JSON.
- D. Terraform is a mutable, procedural, Infrastructure as Code provisioning language based on Hashicorp Configuration Language, or optionally YAML.

**Answer: A ([LEAVE A REPLY](#))**

Terraform is not a configuration management tool -

<https://www.terraform.io/intro/vs/chefpuppet.html> Terraform is a declarative language -

<https://www.terraform.io/docs/configuration/index.html> Terraform supports a syntax that is JSON compatible - <https://www.terraform.io/docs/configuration/syntax-json.html> Terraform is primarily designed on immutable infrastructure principles -

<https://www.hashicorp.com/resources/what-is-mutable-vs-immutable-infrastructure>

### **NEW QUESTION: 45**

The terraform init 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.

- A. False
- B. True

## **Answer: B ([LEAVE A REPLY](#))**

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

## **NEW QUESTION: 46**

Which of the below options is a valid interpolation syntax for retrieving a data source?

- A. \${google\_storage\_bucket.backend}
- B. \${azurerm\_resource\_group.test.data}
- C. \${aws\_instance.web.id.data}
- D. \${data.google\_dns\_keys.foo\_dns\_keys.key\_signing\_keys[0].ds\_record}

## **Answer: ([SHOW ANSWER](#))**

Data source attributes are interpolated with the general syntax

data.TYPENAME.ATTRIBUTE. The interpolation for a resource is the same but without the data. prefix (TYPENAME.ATTRIBUTE).

<https://www.terraform.io/docs/configuration-0-11/interpolation.html#attributes-of-a-data-source>

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## **NEW QUESTION: 47**

1. resource "aws\_s3\_bucket" "example" {
2. bucket = "my-test-s3-terraform-bucket"
3. ...} resource "aws\_iam\_role" "test\_role" {
4. name = "test\_role"
5. ...}

Due to the way that the application code is written, the s3 bucket must be created before the test role is created, otherwise there will be a problem. How can you ensure that?

- A. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- B. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.
- C. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- D. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

### **Answer: A ([LEAVE A REPLY](#))**

Implicit dependency works only if there is some reference of one resource to another.

Explicit dependency is the option here.

### **NEW QUESTION: 48**

Your team has started using terraform OSS in a big way , and now wants to deploy multi region deployments (DR) in aws using the same terraform files . You want to deploy the same infra (VPC,EC2 ...) in both us-east-1 ,and us-west-2 using the same script , and then peer the VPCs across both the regions to enable DR traffic. But , when you run your script , all resources are getting created in only the default provider region. What should you do? Your provider setting is as below -

# The default provider configuration provider "aws" { region = "us-east-1" }

- A.** No way to enable this via a single script . Write 2 different scripts with different default providers in the 2 scripts , one for us-east , another for us-west.
- B.** Create a list of regions , and then use a for-each to iterate over the regions , and create the same resources ,one after the one , over the loop.
- C.** Use provider alias functionality , and add another provider for us-west region . While creating the resources using the tf script , reference the appropriate provider (using the alias).
- D.** Manually create the DR region , once the Primary has been created , since you are using terraform OSS , and multi region deployment is only available in Terraform Enterprise.

### **Answer: C ([LEAVE A REPLY](#))**

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

# The default provider configuration

```
provider "aws" {
```

```
region = "us-east-1"
```

```
}
```

# Additional provider configuration for west coast region

```
provider "aws" {
```

```
alias = "west"
```

```
region = "us-west-2"
```

```
}
```

<https://www.terraform.io/docs/configuration/providers.html>

## **NEW QUESTION: 49**

A variable az has the following default value. What will be the datatype of the variable?

az=["us-west-1a","us-east-1a"]

- A.** String
- B.** Map
- C.** Object
- D.** List

**Answer:** D ([LEAVE A REPLY](#))

## **NEW QUESTION: 50**

Matt wants to import a manually created EC2 instance into terraform so that he can manage the EC2 instance through terraform going forward. He has written the configuration file of the EC2 instance before importing it to Terraform. Following is the code:

```
resource "aws_instance" "matt_ec2" { ami = "ami-bg2640de" instance_type = "t2.micro"  
vpc_security_group_ids = ["sg-6ae7d613", "sg-53370035"] key_name = "mysecret"  
subnet_id = "subnet-9e3cfbc5" } The instance id of that EC2 instance is  
i-0260835eb7e9bd40 How he can import data of EC2 to state file?
```

- A.** terraform import aws\_instance.id = i-0260835eb7e9bd40
- B.** terraform import i-0260835eb7e9bd40
- C.** terraform import aws\_instance.i-0260835eb7e9bd40
- D.** terraform import aws\_instance.matt\_ec2 i-0260835eb7e9bd40

**Answer:** D ([LEAVE A REPLY](#))

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

## **NEW QUESTION: 51**

What is the result of the following terraform function call?

- A.** goodbye

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

- B.** what?

- C.** hello

**Answer:** ([SHOW ANSWER](#))

## **NEW QUESTION: 52**

Hanah is writing a terraform configuration with nested modules, there are multiple places where she has to use the same conditional expression but she wants to avoid repeating the same values or expressions multiple times in the configuration,. What is a better approach to dealing with this?

- A.** Expressions
- B.** Local Values

**C.** Variables

**D.** Functions

**Answer:** ([SHOW ANSWER](#))

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

### **NEW QUESTION: 53**

Named workspaces are not a suitable isolation mechanism for strong separation between staging and production?

**A.** True

**B.** False

**Answer:** ([SHOW ANSWER](#))

Organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

### **NEW QUESTION: 54**

True or False: A list(...) contain a number of values of the same type while an object(...) can contain a number of values of different types.

**A.** False

**B.** True

**Answer:** **B** ([LEAVE A REPLY](#))

Collection Types

A collection type allows multiple values of one other type to be grouped together as a single value. The type of value within a collection is called its element type. All collection types must have an element type, which is provided as the argument to their constructor. For example, the type `list(string)` means "list of strings", which is a different type than `list(number)`, a list of numbers. All elements of a collection must always be of the same type.

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.

\* set(...): a collection of unique values that do not have any secondary identifiers or ordering.

<https://www.terraform.io/docs/configuration/types.html>

### Structural Types

A structural type allows multiple values of several distinct types to be grouped together as a single value. Structural types require a schema as an argument, to specify which types are allowed for which elements.

The two kinds of structural type in the Terraform language are:

\* object(...): a collection of named attributes that each have their own type.

The schema for object types is { <KEY> = <TYPE>, <KEY> = <TYPE>, ... } - a pair of curly braces containing a comma-separated series of <KEY> = <TYPE> pairs. Values that match the object type must contain all of the specified keys, and the value for each key must match its specified type. (Values with additional keys can still match an object type, but the extra attributes are discarded during type conversion.)

\* tuple(...): a sequence of elements identified by consecutive whole numbers starting with zero, where each element has its own type.

The schema for tuple types is [<TYPE>, <TYPE>, ...] - a pair of square brackets containing a comma-separated series of types. Values that match the tuple type must have exactly the same number of elements (no more and no fewer), and the value in each position must match the specified type for that position.

For example: an object type of object({ name=string, age=number }) would match a value like the following:

```
{  
  name = "John"  
  age = 52  
}
```

Also, an object type of object({ id=string, cidr\_block=string }) would match the object produced by a reference to an aws\_vpc resource, like aws\_vpc.example\_vpc; although the resource has additional attributes, they would be discarded during type conversion.

Finally, a tuple type of tuple([string, number, bool]) would match a value like the following:

```
["a", 15, true]
```

<https://www.terraform.io/docs/configuration/types.html>

## NEW QUESTION: 55

If you enable TF\_LOG = DEBUG, the log will be stored in syslog.log file in the current directory.

A. False

B. True

**Answer: A ([LEAVE A REPLY](#))**

<https://www.terraform.io/docs/internals/debugging.html>

## NEW QUESTION: 56

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A. TF\_TRACE
- B. TF\_DEBUG
- C. TF\_LOG
- D. TF\_INFO

**Answer: C ([LEAVE A REPLY](#))**

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

## NEW QUESTION: 57

When multiple engineers start deploying infrastructure using the same state file, what is a feature of remote state storage that is critical to ensure the state doesn't become corrupt?

- A. Object Storage
- B. State Locking
- C. WorkSpaces
- D. Encryption

**Answer: B ([LEAVE A REPLY](#))**

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.

State locking happens automatically on all operations that could write state. You won't see any message that it is happening. If state locking fails, Terraform will not continue. You can disable state locking for most commands with the -lock flag but it is not recommended.

If acquiring the lock is taking longer than expected, Terraform will output a status message. If Terraform doesn't output a message, state locking is still occurring if your backend supports it.

Not all backends support locking. Please view the list of backend types for details on whether a backend supports locking or not.

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

## NEW QUESTION: 58

Your manager has instructed you to start using terraform for the entire infra provisioning of the application stack. There are 4 environments - DEV , QA , UAT , and PROD. The application team has asked for complete segregation between these environments including the backend , state , and also configurations ,since there will be unique resources

in different environments . What is the possible way to structure the terraform code to facilitate that.

- A.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to a different backend.
- B.** Completely separate the working directories , keep one for each environment . For each working directory , maintain a separate configuration file , variables file , and map to the same backend.
- C.** Implement terraform workspaces , and map each environment with one workspace.
- D.** Enable remote backend storage . Configure 4 different backend storages , one for each environment.

**Answer:** A ([LEAVE A REPLY](#))

In particular, organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

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

## NEW QUESTION: 59

What is the default backend for Terraform?

- A.** consul
- B.** gcs
- C.** local
- D.** etcd

**Answer:** ([SHOW ANSWER](#))

By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to.

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

## NEW QUESTION: 60

How does Terraform handle working with so many providers?

- A.** Terraform ships with all of the plugins embedded in the Terraform binary.
- B.** Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in the configuration's working directory.
- C.** Terraform uses a plugin architecture for providers and only installs the provider plugins required by your configuration in a shared, system-wide plugins directory.
- D.** Terraform allows you to select the providers you want to support during the Terraform installation process.

**Answer:** ([SHOW ANSWER](#))

Terraform is built on a plugin-based architecture. All providers and provisioners that are used in Terraform configurations are plugins, even the core types such as AWS and Heroku. Users of Terraform are able to write new plugins in order to support new functionality in Terraform.

#### **NEW QUESTION: 61**

You wanted to destroy some of the dependent resources from real infrastructure. You choose to delete those resources from your configuration file and run terraform plan and then apply. Which of the following way your resources would be destroyed?

- A.** Terraform can still determine the correct order for destruction from the state even when you delete one or more items from the configuration.
- B.** Those would be destroyed in the order in which they were written in the configuration file previously before you have deleted them from configuration file.
- C.** The resource will be destructed in random order as you have already deleted them from configuration.
- D.** You can not destroy resources by deleting them from configuration file and running plan and apply.

#### **Answer: ([SHOW ANSWER](#))**

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state. Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**  
get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

#### **NEW QUESTION: 62**

Which of the following state management command allow you to retrieve a list of resources that are part of the state file?

- A.** terraform state list
- B.** terraform state view

C. terraform view

D. terraform list

**Answer: ([SHOW ANSWER](#))**

The terraform state list command is used to list resources within a Terraform state.

Usage: `terraform state list [options] [address...]`

The command will list all resources in the state file matching the given addresses (if any). If no addresses are given, all resources are listed.

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

### **NEW QUESTION: 63**

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)

- A. Explicit use of a provider block in configuration, optionally including a version constraint.
- B. Existence of any resource instance belonging to a particular provider in the current state.
- C. Existence of any provider plugins found locally in the working directory.

Explanation

The existence of a provider plugin found locally in the working directory does not itself create a provider dependency. The plugin can exist without any reference to it in the terraform configuration. <https://www.terraform.io/docs/commands/providers.html>

D. Use of any resource belonging to a particular provider in a resource or data block in configuration.

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 64**

While using generic git repository as a module source, which of the below options allows terraform to select a specific version or tag instead of selecting the HEAD.

A. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?ref=v1.2.0"}
```

B. Append version argument as

```
module "vpc" { source = "git::https://example.com/vpc.git?version=v1.2.0"}
```

C. Append ref argument as

```
module "vpc" { source = "git::https://example.com/vpc.git#ref=v1.2.0"}
```

D. By default, Terraform will clone and use the default branch (referenced by HEAD) in the selected repository and you can not override this.

**Answer: ([SHOW ANSWER](#))**

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:

```
module "vpc" {  
  source = "git::https://example.com/vpc.git?ref=v1.2.0"  
}
```

The value of the ref argument can be any reference that would be accepted by the git checkout command, including branch and tag names.

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

### **NEW QUESTION: 65**

Which of the following actions are performed during a terraform init?

- A. Initializes downloaded and/or installed providers
- B. Initializes the backend configuration
- C. Provisions the declared resources in your configuration
- D. Download the declared providers which are supported by HashiCorp

**Answer: ([SHOW ANSWER](#))**

The terraform init command is used to initialize a working directory containing Terraform configuration files. This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times.

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.

terraform init command does -

- \* Copy a Source Module
- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

### **NEW QUESTION: 66**

Which of the following commands will launch the Interactive console for Terraform interpolations?

- A. `terraform cmdline`

Explanation

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

- B. `terraform`

- C. `terraform console`

- D. `terraform cli`

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 67**

After running into issues with Terraform, you need to enable verbose logging to assist with troubleshooting the error. Which of the following values provides the MOST verbose logging?

- A. `ERROR`

- B. INFO**
- C. WARN**
- D. TRACE**
- E. DEBUG**

**Answer: D ([LEAVE A REPLY](#))**

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

Examples:

```
export TF_LOG=DEBUG  
export TF_LOG=TRACE
```

### **NEW QUESTION: 68**

What are some of the features of Terraform state? (select three)

- A. increased performance**
- B. determining the correct order to destroy resources**
- C. mapping configuration to real-world resources**
- D. inspection of cloud resources**

**Answer: A,C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 69**

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

- A. False**
- B. True**

**Answer: ([SHOW ANSWER](#))**

You can change your backend configuration at any time. You can change both the configuration itself as well as the type of backend (for example from "consul" to "s3"). Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

### **NEW QUESTION: 70**

terrafrom state subcommands such as list are read-only commands, do read-only commands create state backup files?

- A. Yes**
- B. No**

### **Answer: B ([LEAVE A REPLY](#))**

Subcommands that are read-only (such as list) do not write any backup files since they aren't modifying the state.

All terraform state subcommands that modify the state write backup files. The path of these backup file can be controlled with -backup.

<https://www.terraform.io/docs/commands/state/index.html#backups>

### **NEW QUESTION: 71**

Anyone can publish and share modules on the Terraform Public Module Registry, and meeting the requirements for publishing a module is extremely easy. Select from the following list all valid requirements. (select three)

- A. Module repositories must use this three-part name format, terraform-- .
- B. The registry uses tags to identify module versions.
- C. The module must be on GitHub and must be a public repo.

Explanation

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

- D. The module must be PCI/HIPPA compliant.
- E. Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .

### **Answer: B,C,E ([LEAVE A REPLY](#))**

### **NEW QUESTION: 72**

Which of the following value will be accepted for var1?

```
variable "var1" {  
  type = string  
}
```

- A. None of the above
- B. Both A and B
- C. "5"
- D. 5

### **Answer: B ([LEAVE A REPLY](#))**

Terraform automatically converts number and bool values to strings when needed.

### **NEW QUESTION: 73**

How can you ensure that the engineering team who has access to git repo will not create any non-compliant resources that might lead to a security audit failure in future. your team is using Hashicorp Terraform Enterprise Edition.

- A. Use Terraform OSS Sentinel Lite version , which will save cost , since there is no charge for OSS , but it can still check for most non-compliant rules using Policy-As-Code.
- B. Implement a review process where every code will be reviewed before merging to the master branch.

**C.** Since your team is using Hashicorp Terraform Enterprise Edition , enable Sentinel , and write Policy-As-Code rules that will check for non-compliant resource provisioning , and prevent/report them.

**D.** Create a design /security document (in PDF) and share to the team , and ask them to always follow that document , and never deviate from it.

**Answer:** ([SHOW ANSWER](#))

<https://www.terraform.io/docs/cloud/sentinel/index.html>

## **NEW QUESTION: 74**

lookup retrieves the value of a single element from which of the below data type?

- A.** map
- B.** set
- C.** string
- D.** list

**Answer:** **A** ([LEAVE A REPLY](#))

<https://www.terraform.io/docs/configuration/functions/lookup.html>

## **NEW QUESTION: 75**

What are the benefits of using Infrastructure as Code? (select five)

- A.** Infrastructure as Code is easily repeatable, allowing the user to reuse code to deploy similar, yet different resources
- B.** Infrastructure as Code provides configuration consistency and standardization among deployments
- C.** Infrastructure as Code allows a user to turn a manual task into a simple, automated deployment (Correct) Explanation If you are new to infrastructure as code as a concept, it is the process of managing infrastructure in a file or files rather than manually configuring resources in a user interface.

A resource in this instance is any piece of infrastructure in a given environment, such as a virtual machine, security group, network interface, etc. At a high level, Terraform allows operators to use HCL to author files containing definitions of their desired resources on almost any provider (AWS, GCP, GitHub, Docker, etc) and automates the creation of those resources at the time of application.

- D.** Infrastructure as Code is relatively simple to learn and write, regardless of a user's prior experience with developing code
- E.** Infrastructure as Code gives the user the ability to recreate an application's infrastructure for disaster recovery scenarios
- F.** Infrastructure as Code easily replaces development languages such as Go and .Net for application development

**Answer:** ([SHOW ANSWER](#))

## **NEW QUESTION: 76**

ABC Enterprise has recently tied up with multiple small organizations for exchanging database information. Due to this, the firewall rules are increasing and are more than 100 rules. This is leading firewall configuration file that is difficult to manage. What is the way this type of configuration can be managed easily?

- A. Dynamic Blocks
- B. Terraform Backends
- C. Terraform Functions
- D. Terraform Expression

**Answer: A ([LEAVE A REPLY](#))**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:  
<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**  
**Special Discount: freecram**)

#### **NEW QUESTION: 77**

Which feature of Terraform allows multiple state files for a single configuration file depending upon the environment?

- A. Terraform Remote Backends
- B. Terraform Workspaces
- C. Terraform Enterprise
- D. Terraform Modules

**Answer: B ([LEAVE A REPLY](#))**

#### **NEW QUESTION: 78**

You want to use terraform import to start managing infrastructure that was not originally provisioned through infrastructure as code. Before you can import the resource's current state, what must you do in order to prepare to manage these resources using Terraform?

- A. Run terraform refresh to ensure that the state file has the latest information for existing resources.
- B. Update the configuration file to include the new resources.
- C. Shut down or stop using the resources being imported so no changes are inadvertently missed.
- D. Modify the Terraform state file to add the new resources.

**Answer: B ([LEAVE A REPLY](#))**

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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. The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import  
successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (**324 Q&As Dumps, 30%OFF**

**Special Discount: freecram)**

# HashiCorp.TA-002-P.v2020-10-08.q61

<b>Exam Code:</b>	TA-002-P
<b>Exam Name:</b>	HashiCorp Certified: Terraform Associate
<b>Certification Provider:</b>	HashiCorp
<b>Free Question Number:</b>	61
<b>Version:</b>	v2020-10-08
<b># of views:</b>	2164
<b># of Questions views:</b>	53118

<https://www.freecram.com/torrent/HashiCorp.TA-002-P.v2020-10-08.q61.html>

## NEW QUESTION: 1

A colleague has informed you that a new version of a Terraform module that your team hosts on an Amazon S3 bucket is broken. The Amazon S3 bucket has versioning enabled. Your colleague tells you to make sure you are not using the latest version in your configuration. You have the following configuration block in your code that refers to the module:

module "infranet" { source = "s3::https://s3-us-west-2.amazonaws.com/infrabucket/infra\_module.zip"} What is the best way to ensure that you are not using the latest version of the module?

- A. Add a module version constraint in your configuration's backend block and specify a previous version.
- B. Add a version key to the module configuration and specify a previous version.
- C. Delete the latest version of the module in S3 to rollback to the previous version.
- D. Add a version property to the module in Terraform's state file and specify a previous version.

**Answer:** ([SHOW ANSWER](#))

Explanation

Version constraints are supported only for modules installed from a module registry, such as the 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.

Only Terraform Registries support module versioning by using the version key, one cannot configure a previous version of the module in the configuration. Deleting the latest version of the module in S3 is the only option of the available options that ensures you won't use the latest version. You could also modify the source URL to specify a versionId URL parameter for a previous version.

<https://www.terraform.io/docs/configuration/modules.html#source>

### **NEW QUESTION: 2**

What kind of resource dependency is stored in terraform.tfstate file?

- A. Both implicit and explicit dependencies are stored in state file.
- B. Only explicit dependencies are stored in state file.
- C. Only implicit dependencies are stored in state file.
- D. No dependency information is stored in state file.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 3**

What does terraform refresh command do?

- A. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- B. terraform refresh is used to change/modify the infrastructure based on the existing state file, at that moment.
- C. terraform refresh can be used to selectively update sections of the state file, using terraform resource level addressing.
- D. terraform refresh syncs the state file with the real world infrastructure.
- E. terraform refresh command basically updates the configuration file with the current state of the actual infrastructure

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 4**

You have declared a variable name my\_var in terraform configuration without a value associated with it.

variable my\_var {}

After running terraform plan it will show an error as variable is not defined.

- A. True
- B. False

**Answer: ([SHOW ANSWER](#))**

Explanation

Input variables are usually defined by stating a name, type and a default value. However, the type and default values are not strictly necessary. Terraform can deduct the type of the variable from the default or input value.

Variables can be predetermined in a file or included in the command-line options. As such, the simplest variable is just a name while the type and value are selected based on the input.

```
variable "variable_name" {}  
terraform apply -var variable_name="value"
```

The input variables, like the one above, use a couple of different types: strings, lists, maps, and boolean. Here are some examples of how each type are defined and used.

### String

Strings mark a single value per structure and are commonly used to simplify and make complicated values more user-friendly. Below is an example of a string variable definition.

```
variable "template" {  
  type = string  
  default = "01000000-0000-4000-8000-000030080200"  
}
```

A string variable can then be used in resource plans. Surrounded by double quotes, string variables are a simple substitution such as the example underneath.

```
storage = var.template
```

### List

Another type of Terraform variables lists. They work much like a numbered catalogue of values. Each value can be called by their corresponding index in the list. Here is an example of a list variable definition.

```
variable "users" {  
  type = list  
  default = ["root", "user1", "user2"]  
}
```

Lists can be used in the resource plans similarly to strings, but you'll also need to denote the index of the value you are looking for.

```
username = var.users[0]
```

### Map

Maps are a collection of string keys and string values. These can be useful for selecting values based on predefined parameters such as the server configuration by the monthly price.

```
variable "plans" {  
  type = map  
  default = {  
    "5USD" = "1xCPU-1GB"  
    "10USD" = "1xCPU-2GB"  
    "20USD" = "2xCPU-4GB"
```

```
}
```

```
}
```

You can access the right value by using the matching key. For example, the variable below would set the plan to "1xCPU-1GB".

```
plan = var.plans["5USD"]
```

The values matching to their keys can also be used to look up information in other maps.

For example, underneath is a shortlist of plans and their corresponding storage sizes.

```
variable "storage_sizes" {
```

```
  type = map
```

```
  default = {
```

```
    "1xCPU-1GB" = "25"
```

```
    "1xCPU-2GB" = "50"
```

```
    "2xCPU-4GB" = "80"
```

```
}
```

```
}
```

These can then be used to find the right storage size based on the monthly price as defined in the previous example.

```
size = lookup(var.storage_sizes, var.plans["5USD"])
```

Boolean

The last of the available variable type is boolean. They give the option to employ simple true or false values.

For example, you might wish to have a variable that decides when to generate the root user password on a new deployment.

```
variable "set_password" {
```

```
  default = false
```

```
}
```

The above example boolean can be used similarly to a string variable by simply marking down the correct variable.

```
create_password = var.set_password
```

By default, the value is set to false in this example. However, you can overwrite the variable at deployment by assigning a different value in a command-line variable.

```
terraform apply -var set_password="true"
```

## NEW QUESTION: 5

Complete the following sentence:

For local state, the workspaces are stored directly in a \_\_\_\_\_.

- A. a file called terraform.tfstate.backup
- B. directory called terraform.workspaces.tfstate
- C. a file called terraform.tfstate
- D. directory called terraform.tfstate.d

**Answer: D ([LEAVE A REPLY](#))**

## Explanation

For local state, Terraform stores the workspace states in a directory called `terraform.tfstate.d`.

<https://www.terraform.io/docs/state/workspaces.html#workspace-internals>

## NEW QUESTION: 6

Once a resource is marked as tainted, the next plan will show that the resource will be \_\_\_\_\_ and \_\_\_\_\_

\_\_\_\_\_ and the next apply will implement this change.

- A. destroyed and recreated
- B. destroyed and not recreated
- C. tainted and not destroyed
- D. recreated and tainted

**Answer:** A ([LEAVE A REPLY](#))

## NEW QUESTION: 7

You have created a terraform script that uses a lot of new constructs that have been introduced in terraform v0.12. However, many developers who are cloning the script from your git repo, are using v0.11, and getting errors. What can be done from your end to solve this problem?

- A. Force developer to use v0.12 by using terraform setting '`required_version`' and set it to `>=0.12`.
- B. Refactor the code to support both v0.11, and v0.12. It might be a difficult process, but there is no other way.
- C. Add a condition in front of each such specific construct, to check whether the running terraform version is v0.11 or v0.12, and work accordingly.
- D. Add comments in your code to tell developers to use v0.12 . If they use v0.11 , that should be their problem , which they need to figure out.

**Answer:** ([SHOW ANSWER](#))

## Explanation

<https://www.terraform.io/docs/configuration/terraform.html>

## NEW QUESTION: 8

During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?

- A. The resource will be planned for destruction and recreation upon the next terraform apply
- B. Terraform will retry to provision again.
- C. The failure of provisioner will be ignored and it will not cause a failure to terraform apply
- D. The resource will be automatically destroyed.

**Answer:** A ([LEAVE A REPLY](#))

## Explanation

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 cannot reason about what the provisioner does, the only way to ensure proper creation of a resource is to recreate it. This is tainting.

You can change this behavior by setting the `on_failure` attribute, which is covered in detail below.

<https://www.terraform.io/docs/provisioners/index.html#creation-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#destroy-time-provisioners>

<https://www.terraform.io/docs/provisioners/index.html#failure-behavior>

## NEW QUESTION: 9

What is a downside to using the Vault provider to read secrets from Vault?

- A. Secrets are persisted to the state file and plans.
- B. Terraform and Vault must be running on the same version.
- C. Terraform and Vault must be running on the same physical host.
- D. Terraform requires a unique auth method to work with Vault.

**Answer: A ([LEAVE A REPLY](#))**

## Explanation

The Vault provider allows Terraform to read from, write to, and configure Hashicorp Vault. Interacting with Vault from Terraform causes any secrets that you read and write to be persisted in both Terraform's state file and in any generated plan files. For any Terraform module that reads or writes Vault secrets, these files should be treated as sensitive and protected accordingly.

## NEW QUESTION: 10

A user has created three workspaces using the command line - prod, dev, and test. The user wants to create a fourth workspace named stage. Which command will the user execute to accomplish this?

- A. `terraform workspace new stage`
- B. `terraform workspace -new stage`
- C. `terraform workspace -create stage`
- D. `terraform workspace create stage`

**Answer: A ([LEAVE A REPLY](#))**

## Explanation

The `terraform workspace new` command is used to create a new workspace.

<https://www.terraform.io/docs/commands/workspace/new.html>

## NEW QUESTION: 11

Which of the following Terraform commands will automatically refresh the state unless supplied with additional flags or arguments? Choose TWO correct answers.

- A. `terraform validate`
- B. `terraform output`
- C. `terraform apply`
- D. `terraform plan`
- E. `terraform state`

**Answer: C,D** ([LEAVE A REPLY](#))

### **NEW QUESTION: 12**

Terraform must track metadata such as resource dependencies. Where is this data stored?

- A. workspace
- B. backend
- C. state file
- D. metadata store

**Answer: C** ([LEAVE A REPLY](#))

Explanation

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

<https://www.terraform.io/docs/state/purpose.html#metadata>

### **NEW QUESTION: 13**

Which of the below datatype is not supported by Terraform.

- A. List
- B. Map
- C. Array
- D. Object

**Answer: C** ([LEAVE A REPLY](#))

### **NEW QUESTION: 14**

Which of the below backends support state locking?

- A. azurerm
- B. consul

C. S3

D. artifactory

**Answer: A,B,C ([LEAVE A REPLY](#))**

### **NEW QUESTION: 15**

Command terraform refresh will update state file?

A. False

B. True

**Answer: B ([LEAVE A REPLY](#))**

Explanation

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. If the state is changed, this may cause changes to occur during the next plan or apply.

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

### **NEW QUESTION: 16**

You wanted to destroy some of the dependent resources from real infrastructure. You choose to delete those resources from your configuration file and run terraform plan and then apply. Which of the following way your resources would be destroyed?

A. Terraform can still determine the correct order for destruction from the state even when you delete one or more items from the configuration.

B. Those would be destroyed in the order in which they were written in the configuration file previously before you have deleted them from configuration file.

C. The resource will be destructed in random order as you have already deleted them from configuration.

D. You can not destroy resources by deleting them from configuration file and running plan and apply.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Terraform typically uses the configuration to determine dependency order. However, when you delete a resource from a Terraform configuration, Terraform must know how to delete that resource. Terraform can see that a mapping exists for a resource not in your configuration and plan to destroy. However, since the configuration no longer exists, the order cannot be determined from the configuration alone.

To ensure correct operation, Terraform retains a copy of the most recent set of dependencies within the state.

Now Terraform can still determine the correct order for destruction from the state when you delete one or more items from the configuration.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam! Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here: <https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF** **Special Discount: freecram**)

### **NEW QUESTION: 17**

Which of the following clouds does not have a provider maintained HashiCorp?

- A. IBM Cloud
- B. DigitalOcean
- C. OpenStack
- D. AWS

**Answer: A ([LEAVE A REPLY](#))**

Explanation

IBM Cloud does not have a provider maintained by HashiCorp, although IBM Cloud does maintain their own Terraform provider.

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

### **NEW QUESTION: 18**

Your organization has moved to AWS and has manually deployed infrastructure using the console. Recently, a decision has been made to standardize on Terraform for all deployments moving forward.

What can you do to ensure that all existing is managed by Terraform moving forward without interruption to existing services?

- A. Submit a ticket to AWS and ask them to export the state of all existing resources and use terraform import to import them into the state file.
- B. Delete the existing resources and recreate them using new a Terraform configuration so Terraform can manage them moving forward.
- C. Resources that are manually deployed in the AWS console cannot be imported by Terraform.
- D. Using terraform import, import the existing infrastructure into your Terraform state.

**Answer: D ([LEAVE A REPLY](#))**

Explanation

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The `terraform import` command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

## NEW QUESTION: 19

Which Terraform command will check and report errors within modules, attribute names, and value types to make sure they are syntactically valid and internally consistent?

- A. `terraform validate`
- B. `terraform format`
- C. `terraform fmt`
- D. `terraform show`

**Answer: ([SHOW ANSWER](#))**

Explanation

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 the 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.

## NEW QUESTION: 20

Named workspaces are not a suitable isolation mechanism for strong separation between staging and production?

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Organizations commonly want to create a strong separation between multiple deployments of the same infrastructure serving different development stages (e.g. staging vs. production) or different internal teams. In this case, the backend used for each deployment often belongs to that deployment, with different credentials and access controls. Named workspaces are not a suitable isolation mechanism for this scenario.

<https://www.terraform.io/docs/state/workspaces.html#when-to-use-multiple-workspaces>

## NEW QUESTION: 21

What resource dependency information is stored in Terraform's state?

- A. Only implicit dependencies are stored in state.
- B. Both implicit and explicit dependencies are stored in state.
- C. Only explicit dependencies are stored in state.
- D. No dependency information is stored in state.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Terraform state captures all dependency information, both implicit and explicit. One purpose for state is to determine the proper order to destroy resources. When resources are created all of their dependency information is stored in the state. If you destroy a resource with dependencies, Terraform can still determine the correct destroy order for all other resources because the dependencies are stored in the state.

<https://www.terraform.io/docs/state/purpose.html#metadata>

## NEW QUESTION: 22

Which of the following actions are performed during a terraform init?

- A. Initializes downloaded and/or installed providers
- B. Initializes the backend configuration
- C. Provisions the declared resources in your configuration
- D. Download the declared providers which are supported by HashiCorp

**Answer: ([SHOW ANSWER](#))**

Explanation

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

This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times.

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.

terraform init command does -

- \* Copy a Source Module
- \* Backend Initialization
- \* Child Module Installation
- \* Plugin Installation

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

### **NEW QUESTION: 23**

Which of the below are paid features of Terraform Cloud?

- A.** Full API Coverage
- B.** Secure variable Storage
- C.** Roles/ Team management
- D.** Cost Estimation
- E.** Private Module Registry
- F.** Sentinel policies

**Answer:** ([SHOW ANSWER](#))

Explanation

<https://www.hashicorp.com/products/terraform/pricing/>

### **NEW QUESTION: 24**

When multiple engineers start deploying infrastructure using the same state file, what is a feature of remote state storage that is critical to ensure the state doesn't become corrupt?

- A.** Object Storage
- B.** State Locking
- C.** WorkSpaces
- D.** Encryption

**Answer:** **B** ([LEAVE A REPLY](#))

Explanation

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.

State locking happens automatically on all operations that could write state. You won't see any message that it is happening. If state locking fails, Terraform will not continue. You can disable state locking for most commands with the `-lock` flag but it is not recommended.

If acquiring the lock is taking longer than expected, Terraform will output a status message. If Terraform doesn't output a message, state locking is still occurring if your backend supports it.

Not all backends support locking. Please view the list of backend types for details on whether a backend supports locking or not.

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

### **NEW QUESTION: 25**

The Security Operations team of ABC Enterprise wants to mandate that all the Terraform configuration that creates an S3 bucket must have encryption feature enabled. What is the best way to achieve it?

- A.** Use Sentinel Policies.
- B.** Use S3 bucket policy.
- C.** Create a script that checks the encryption parameter is enabled on every git commit.
- D.** Shared a SOP to engineers to mandate encryption feature on S3.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Sentinel is an embedded policy-as-code framework integrated with the HashiCorp Enterprise products. It enables fine-grained, logic-based policy decisions, and can be extended to use information from external sources.

Using Sentinel with Terraform Cloud involves:

- \* Defining the policies - Policies are defined using the policy language with imports for parsing the Terraform plan, state and configuration.
- \* Managing policies for organizations - Users with permission to manage policies can add policies to their organization by configuring VCS integration or uploading policy sets through the API. They also define which workspaces the policy sets are checked against during runs. (More about permissions.)
- \* 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.
- \* Mocking Sentinel Terraform data - Terraform Cloud provides the ability to generate mock data for any run within a workspace. This data can be used with the Sentinel CLI to test policies before deployment.

<https://www.terraform.io/docs/cloud/sentinel/index.html>

### **NEW QUESTION: 26**

lookup retrieves the value of a single element from which of the below data type?

- A.** map
- B.** set
- C.** string
- D.** list

**Answer: A ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/configuration/functions/lookup.html>

## **NEW QUESTION: 27**

In regards to deploying resources in multi-cloud environments, what are some of the benefits of using Terraform rather than a provider's native tooling? (select three)

- A.** Terraform can help businesses deploy applications on multiple clouds and on-premises infrastructure.
- B.** Terraform is not cloud-agnostic and can be used to deploy resources across a single public cloud.
- C.** Terraform simplifies management and orchestration, helping operators build large-scale, multi-cloud infrastructure.
- D.** Terraform can manage cross-cloud dependencies.

**Answer: A,C,D ([LEAVE A REPLY](#))**

Explanation

Terraform is cloud-agnostic and allows a single configuration to be used to manage multiple providers, and to even handle cross-cloud dependencies. This simplifies management and orchestration, helping operators build large-scale multi-cloud infrastructures.

<https://www.terraform.io/intro/use-cases.html>

## **NEW QUESTION: 28**

Terraform has detailed logs which can be enabled by setting the \_\_\_\_\_ environmental variable.

- A.** TF\_TRACE
- B.** TF\_DEBUG
- C.** TF\_LOG
- D.** TF\_INFO

**Answer: C ([LEAVE A REPLY](#))**

Explanation

Terraform has detailed logs that 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. TRACE is the most verbose and it is the default if TF\_LOG is set to something other than a log level name.

<https://www.terraform.io/docs/internals/debugging.html>

## **NEW QUESTION: 29**

What is the command you can use to set an environment variable named "var1" of type String?

- A.** export TF\_VAR\_VAR1
- B.** set TF\_VAR\_var1
- C.** variable "var1" { type = "string"}

D. export TF\_VAR\_var1

**Answer: D ([LEAVE A REPLY](#))**

Explanation

The environment variable must be in the format TF\_VAR\_name, so for the question TF\_VAR\_var1 is the correct choice.

[https://www.terraform.io/docs/commands/environment-variables.html#tf\\_var\\_name](https://www.terraform.io/docs/commands/environment-variables.html#tf_var_name)

### **NEW QUESTION: 30**

A variable az has the following default value. What will be the datatype of the variable?

az=["us-west-1a","us-east-1a"]

- A. Object
- B. Map
- C. String
- D. List

**Answer: ([SHOW ANSWER](#))**

### **NEW QUESTION: 31**

In regards to Terraform state file, select all the statements below which are correct?

- A. When using local state, the state file is stored in plain-text.
- B. The state file is always encrypted at rest.
- C. Storing state remotely can provide better security.
- D. Using the mask feature, you can instruct Terraform to mask sensitive data in the state file.
- E. The Terraform state can contain sensitive data, therefore the state file should be protected from unauthorized access.
- F. Terraform Cloud always encrypts state at rest.

**Answer: ([SHOW ANSWER](#))**

Explanation

Terraform state can contain sensitive data, depending on the resources in use and your definition of

"sensitive." The state contains resource IDs and all resource attributes. For resources such as databases, this may contain initial passwords.

When using local state, state is stored in plain-text JSON files.

When using remote state, state is only ever held in memory when used by Terraform. It may be encrypted at rest, but this depends on the specific remote state backend.

Storing Terraform 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.

Recommendations

If you manage any sensitive data with Terraform (like database passwords, user passwords, or private keys), treat the state itself as sensitive data.

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.

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

## NEW QUESTION: 32

You cannot publish your own modules on the Terraform Registry.

A. False

B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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

## NEW QUESTION: 33

Valarie has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code. Valarie wants to hide the output value in the CLI after terraform apply that's why she has used sensitive parameter.

1. output "db\_password" {
2. value = local.db\_password
3. sensitive = true
4. }

Since sensitive is set to true, will the value associated with db password be available in plain-text in the state file for everyone to read?

A. Yes

B. No

## **Answer: A ([LEAVE A REPLY](#))**

### Explanation

Outputs can be marked as containing sensitive material by setting the sensitive attribute to true, like this:

```
output "sensitive" {  
    sensitive = true  
    value = VALUE  
}
```

When outputs are displayed on-screen following a terraform apply or terraform refresh, sensitive outputs are redacted, with <sensitive> displayed in place of their value.

### Limitations of Sensitive Outputs

The values of sensitive outputs are still stored in the Terraform state, and available using the terraform output command, so cannot be relied on as a sole means of protecting values.

Sensitivity is not tracked internally, so if the output is interpolated in another module into a resource, the value will be displayed.

## **NEW QUESTION: 34**

Mary has created a database instance in AWS and for ease of use is outputting the value of the database password with the following code:

1. output "db\_password"
2. {
3. value = local.db\_password
4. }

Mary wants to hide the output value in the CLI after terraform apply? What is the best way?

- A. Use cryptographic hash
- B. Use sensitive parameter
- C. Encrypt the value using encrypt() function
- D. Use secure parameter

## **Answer: B ([LEAVE A REPLY](#))**

## **NEW QUESTION: 35**

Which of the below options is a valid interpolation syntax for retrieving a data source?

- A. \${google\_storage\_bucket.backend}
- B. \${azurerm\_resource\_group.test.data}
- C. \${aws\_instance.web.id.data}
- D. \${data.google\_dns\_keys.foo\_dns\_keys.key\_signing\_keys[0].ds\_record}

## **Answer: D ([LEAVE A REPLY](#))**

### Explanation

Data source attributes are interpolated with the general syntax `data.TYPE.NAME.ATTRIBUTE`. The interpolation for a resource is the same but without the `data.` prefix (`TYPE.NAME.ATTRIBUTE`).

<https://www.terraform.io/docs/configuration-0-11/interpolation.html#attributes-of-a-data-source>

## NEW QUESTION: 36

Given the Terraform configuration below, in which order will the resources be created?

```
1. resource "aws_instance" "web_server"
2. {
3.   ami = "ami-b374d5a5"
4.   instance_type = "t2.micro"
5. }
6. resource "aws_eip" "web_server_ip"
7. {
8.   vpc = true
9.   instance = aws_instance.web_server.id
}
```

- A. `aws_eip` will be created first  
`aws_instance` will be created second
- B. `aws_eip` will be created first  
`aws_instance` will be created second
- C. Resources will be created simultaneously
- D. `aws_instance` will be created first  
`aws_eip` will be created second

**Answer: D (LEAVE A REPLY)**

Explanation

Implicit and Explicit Dependencies

By studying the resource attributes used in interpolation expressions, Terraform can automatically infer when one resource depends on another. In the example above, the reference to `aws_instance.web_server.id` creates an implicit dependency on the `aws_instance` named `web_server`.

Terraform uses this dependency information to determine the correct order in which to create the different resources.

```
# Example of Implicit Dependency
resource "aws_instance" "web_server" {
  ami = "ami-b374d5a5"
  instance_type = "t2.micro"
}
resource "aws_eip" "web_server_ip" {
  vpc = true
  instance = aws_instance.web_server.id
}
```

```
}
```

In the example above, Terraform knows that the `aws_instance` must be created before the `aws_eip`.

Implicit dependencies via interpolation expressions are the primary way to inform Terraform about these relationships, and should be used whenever possible.

Sometimes there are dependencies between resources that are not visible to Terraform. The `depends_on` argument is accepted by any resource and accepts a list of resources to create explicit dependencies for.

For example, perhaps an application we will run on our EC2 instance expects to use a specific Amazon S3 bucket, but that dependency is configured inside the application code and thus not visible to Terraform. In that case, we can use `depends_on` to explicitly declare the dependency:

```
# Example of Explicit Dependency
```

```
# New resource for the S3 bucket our application will use.
```

```
resource "aws_s3_bucket" "example" {  
  bucket = "terraform-getting-started-guide"  
  acl = "private"  
}
```

```
# Change the aws_instance we declared earlier to now include "depends_on" resource  
"aws_instance" "example" { ami = "ami-2757f631" instance_type = "t2.micro"
```

```
# Tells Terraform that this EC2 instance must be created only after the  
# S3 bucket has been created.
```

```
depends_on = [aws_s3_bucket.example]  
}
```

<https://learn.hashicorp.com/terraform/getting-started/dependencies.html>

## NEW QUESTION: 37

What does terraform plan do ?

- A. Create an execution plan by evaluating the difference between configuration file and actual infrastructure.
- B. Create an execution plan by evaluating the difference between configuration file and state file.
- C. Performs a refresh, unless explicitly disabled, and then apply the changes that are necessary to achieve the desired state specified in the configuration files.
- D. Checks whether the execution plan for a set of changes matches your expectations by making changes to real resources or to the state.

**Answer: B ([LEAVE A REPLY](#))**

## NEW QUESTION: 38

Terraform import command can import resources into modules as well directly into the root of your state.

**A.** True

**B.** False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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 directly into the root of your state.

Terraform is able to import existing infrastructure. This allows us take resources we've created by some other means (i.e. via console) and bring it under Terraform management. This is a great way to slowly transition infrastructure to Terraform.

The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform. For example:

```
resource "aws_instance" "import_example" {  
  # ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration:

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

As a result of the above command, the resource is recorded in the state file. We can now run terraform plan to see how the configuration compares to the imported resource, and make any adjustments to the configuration to align with the current (or desired) state of the imported object.

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

## NEW QUESTION: 39

You are using a terraform operation that writes state. Unfortunately automatic state unlocking has failed for that operation. Which of the below commands can be used to remove the already acquired lock on the state?

- A. terraform unlock
- B. terraform force-unlock
- C. terraform state unlock
- D. None of the above

**Answer: B ([LEAVE A REPLY](#))**

Explanation

Command: force-unlock

Manually unlock the state for the defined configuration.

This will not modify your infrastructure. This command removes the lock on the state for the current configuration. The behavior of this lock is dependent on the backend being used. Local state files cannot be unlocked by another process.

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

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

Terraform has a force-unlock command to manually unlock the state if unlocking failed. 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.

#### **NEW QUESTION: 40**

Terraform works well in Windows but a Windows server is required.

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

You may see this question in actual exam. Please remember : Terraform does not require GO language to be installed as a prerequisite and it does not require a Windows Server as well.

#### **NEW QUESTION: 41**

You have created a custom variable definition file my\_vars.tfvars. How will you use it for provisioning infrastructure?

- A. terraform apply -var-state-file ="my\_vars.tfvars"
- B. terraform apply var-file="my\_vars.tfvars"
- C. terraform plan -var-file="my\_vars.tfvar"
- D. terraform apply -var-file="my\_vars.tfvars"

**Answer: D ([LEAVE A REPLY](#))**

Explanation

To set lots of variables, it is more convenient to specify their values in a variable definitions file (with a filename ending in either .tfvars or .tfvars.json) and then specify that file on the command line with -var-file:

`terraform apply -var-file="my_vars.tfvars"`

<https://www.terraform.io/docs/configuration/variables.html#variable-definitions-tfvars-files>

### **NEW QUESTION: 42**

Your company has a lot of workloads in AWS , and Azure that were respectively created using CloudFormation , and AzureRM Templates. However , now your CIO has decided to use Terraform for all new projects , and has asked you to check how to integrate the existing environment with terraform code.

What should be your next plan of action?

- A. This is only possible in Terraform Enterprise , which has the TerraformConverter exe that can take any other template language like AzureRM and convert to Terraform code.
- B. Tell the CIO that this is not possible . Resources created in CloudFormation , and AzureRM templates cannot be tracked using terraform.
- C. Just write the terraform config file for the new resources , and run terraform apply , the state file will automatically be updated with the details of the new resources to be imported.
- D. Use terraform import command to import each resource one by one .

**Answer: D ([LEAVE A REPLY](#))**

### **NEW QUESTION: 43**

In Terraform Enterprise, a workspace can be mapped to how many VCS repos?

- A. 5
- B. 2
- C. 3
- D. 1

**Answer: D ([LEAVE A REPLY](#))**

Explanation

A workspace can only be configured to a single VCS repo, however, multiple workspaces can use the same repo.

<https://www.terraform.io/docs/cloud/workspaces/vcs.html>

### **NEW QUESTION: 44**

1. resource "aws\_s3\_bucket" "example" {
2. bucket = "my-test-s3-terraform-bucket"
3. ...} resource "aws\_iam\_role" "test\_role" {
4. name = "test\_role"
5. ...}

Due to the way that the application code is written, the s3 bucket must be created before the test role is created, otherwise there will be a problem. How can you ensure that?

- A. Add explicit dependency using depends\_on . This will ensure the correct order of resource creation.
- B. This will already be taken care of by terraform native implicit dependency. Nothing else needs to be done from your end.

- C. This is not possible to control in terraform . Terraform will take care of it in a native way , and create a dependency graph that is best suited for the parallel resource creation.
- D. Create 2 separate terraform config scripts , and run them one by one , 1 for s3 bucket , and another for IAM role , run the S3 bucket script first.

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Implicit dependency works only if there is some reference of one resource to another.  
Explicit dependency is the option here.

### **NEW QUESTION: 45**

You want to use terraform import to start managing infrastructure that was not originally provisioned through infrastructure as code. Before you can import the resource's current state, what must you do in order to prepare to manage these resources using Terraform?

- A. Run terraform refresh to ensure that the state file has the latest information for existing resources.
- B. Update the configuration file to include the new resources.
- C. Shut down or stop using the resources being imported so no changes are inadvertently missed.
- D. Modify the Terraform state file to add the new resources.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The current implementation of Terraform import can only import resources into the state. It does not generate configuration. A future version of 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. The terraform import command is used to import existing infrastructure.

To import a resource, first write a resource block for it in our configuration, establishing the name by which it will be known to Terraform.

Example:

```
resource "aws_instance" "import_example" {  
# ...instance configuration...  
}
```

Now terraform import can be run to attach an existing instance to this resource configuration.

```
$ terraform import aws_instance.import_example i-03efafa258104165f  
aws_instance.import_example: Importing from ID "i-03efafa258104165f"...  
aws_instance.import_example: Import complete!  
Imported aws_instance (ID: i-03efafa258104165f)  
aws_instance.import_example: Refreshing state... (ID: i-03efafa258104165f) Import  
successful!
```

The resources that were imported are shown above. These resources are now in your Terraform state and will henceforth be managed by Terraform.

This command locates the AWS instance with ID i-03efafa258104165f (which has been created outside Terraform) and attaches its existing settings, as described by the EC2 API, to the name aws\_instance.import\_example in the Terraform state.

### NEW QUESTION: 46

By default, a defined provisioner is a creation-time provisioner.

- A. True
- B. False

**Answer: A ([LEAVE A REPLY](#))**

Explanation

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com TA-002-P exam **questions have been updated** and **answers have been corrected** get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

### NEW QUESTION: 47

Which of the following allows Terraform users to apply policy as code to enforce standardized configurations for resources being deployed via infrastructure as code?

- A. Sentinel
- B. Module registry
- C. Functions
- D. Workspaces

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Sentinel is a language and framework for policy built to be embedded in existing software to enable fine-grained, logic-based policy decisions. A policy describes under what circumstances certain behaviors are allowed. Sentinel is an enterprise-only feature.

[https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb\\_title](https://www.youtube.com/watch?v=Vy8s7AAvU6g&feature=emb_title)

### NEW QUESTION: 48

If you delete a remote backend from the configuration, will you need to rebuild your state files locally?

- A. False

**B.** True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

You can change your backend configuration at any time. You can change both the configuration itself as well as the type of backend (for example from "consul" to "s3"). Terraform will automatically detect any changes in your configuration and request a reinitialization. As part of the reinitialization process, Terraform will ask if you'd like to migrate your existing state to the new configuration. This allows you to easily switch from one backend to another.

<https://www.terraform.io/docs/backends/config.html#changing-configuration>

### **NEW QUESTION: 49**

Which of the following command can be used to view the specified version constraints for all providers used in the current configuration.

- A.** terraform providers
- B.** terraform state show
- C.** terraform provider
- D.** terraform plan

**Answer: A ([LEAVE A REPLY](#))**

Explanation

Use the `terraform providers` command to view the specified version constraints for all providers used in the current configuration.

<https://www.terraform.io/docs/configuration/providers.html>

### **NEW QUESTION: 50**

Multiple provider instances blocks for AWS can be part of a single configuration file?

- A.** False
- B.** True

**Answer: ([SHOW ANSWER](#))**

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration. For example:

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}
```

```
# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it creates an additional provider configuration. For providers that have no required configuration arguments, the implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html#alias-multiple-provider-instances>

### **NEW QUESTION: 51**

Multiple providers can be declared within a single Terraform configuration file.

A. True

B. False

**Answer:** ([SHOW ANSWER](#))

Explanation

You can optionally define multiple configurations for the same provider, and select which one to use on a per-resource or per-module basis. The primary reason for this is to support multiple regions for a cloud platform; other examples include targeting multiple Docker hosts, multiple Consul hosts, etc.

To include multiple configurations for a given provider, include multiple provider blocks with the same provider name, but set the alias meta-argument to an alias name to use for each additional configuration.

For Example

```
# The default provider configuration
provider "aws" {
  region = "us-east-1"
}

# Additional provider configuration for west coast region
provider "aws" {
  alias = "west"
  region = "us-west-2"
}
```

The provider block without alias set is known as the default provider configuration. When alias is set, it creates an additional provider configuration. For providers that have no required configuration arguments, the implied empty configuration is considered to be the default provider configuration.

<https://www.terraform.io/docs/configuration/providers.html>

## **NEW QUESTION: 52**

Terraform-specific settings and behaviors are declared in which configuration block type?

- A. provider
- B. terraform
- C. resource
- D. data

**Answer: B ([LEAVE A REPLY](#))**

Explanation

The special terraform configuration block type is used to configure some behaviors of Terraform itself, such as requiring a minimum Terraform version to apply your configuration.

Example

```
terraform {  
    required_version = "> 0.12.0"  
}
```

<https://www.terraform.io/docs/configuration/terraform.html>

## **NEW QUESTION: 53**

Which of the following challenges would Terraform be a candidate for solving? (Select THREE)

- A. Reduce the number of workflows needed for managing infrastructure across each of the companies public and private clouds.
- B. Utilize a single tool for all of the infrastructure and configuration management needs.
- C. Enable self-service infrastructure to allocate resources on your proprietary private cloud.
- D. Have a single interoperable tool to manage the variety of services including GitHub repositories, MySQL database, and Kubernetes clusters.

**Answer: ([SHOW ANSWER](#))**

## **NEW QUESTION: 54**

Using multi-cloud and provider-agnostic tools provides which of the following benefits?

- A. Operations teams only need to learn and manage a single tool to manage infrastructure, regardless of where the infrastructure is deployed.
- B. Increased risk due to all infrastructure relying on a single tool for management.
- C. Can be used across major cloud providers and VM hypervisors.
- D. Slower provisioning speed allows the operations team to catch mistakes before they are applied.

**Answer: A,C ([LEAVE A REPLY](#))**

Explanation

Using a tool like Terraform can be advantageous for organizations deploying workloads across multiple public and private cloud environments. Operations teams only need to

learn a single tool, single language, and can use the same tooling to enable a DevOps-like experience and workflows.

### **NEW QUESTION: 55**

From the answers below, select the advantages of using Infrastructure as Code.

- A.** Provide a codified workflow to develop customer-facing applications.
- B.** Safely test modifications using a "dry run" before applying any actual changes.
- C.** Easily integrate with application workflows (GitLab Actions, Azure DevOps, CI/CD tools).
- D.** Easily change and update existing infrastructure.
- E.** Provide reusable modules for easy sharing and collaboration.

**Answer:** ([SHOW ANSWER](#))

Explanation

Infrastructure as Code is not used to develop applications, but it can be used to help deploy or provision those applications to a public cloud provider or on-premises infrastructure.

All of the others are benefits to using Infrastructure as Code over the traditional way of managing infrastructure, regardless if it's public cloud or on-premises.

### **NEW QUESTION: 56**

What is the default backend for Terraform?

- A.** consul
- B.** gcs
- C.** local
- D.** etcd

**Answer:** C ([LEAVE A REPLY](#))

Explanation

By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to.

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

### **NEW QUESTION: 57**

Which of the following is the right substitute for static values that can make Terraform configuration file more dynamic and reusable?

- A.** Output value
- B.** Input parameters
- C.** Functions
- D.** Modules

**Answer:** ([SHOW ANSWER](#))

Explanation

Input variables serve as parameters for a Terraform module, allowing aspects of the module to be customized without altering the module's own source code, and allowing modules to be shared between different configurations.

### **NEW QUESTION: 58**

You have already set TF\_LOG = DEBUG to enable debug log. Now you want to always write the log to the directory you're currently running terraform from. what should you do to achieve this.

- A. Run the command export TF\_LOG\_FILE=./terraform.log.
- B. Run the command export TF\_LOG\_PATH=./terraform.log.
- C. Run the command export TF\_DEBUG\_PATH=./terraform.log.
- D. No explicit action required. Terraform will take care of this as you have enable TF\_LOG.

**Answer: B ([LEAVE A REPLY](#))**

Explanation

<https://www.terraform.io/docs/commands/environment-variables.html>

### **NEW QUESTION: 59**

You have created a custom variable definition file testing.tfvars. How will you use it for provisioning infrastructure?

- A. terraform apply -var-state-file ="testing.tfvars"
- B. terraform plan -var-file="testing.tfvar"
- C. terraform apply -var-file="testing.tfvars"
- D. terraform apply var-file="testing.tfvars"

**Answer: C ([LEAVE A REPLY](#))**

Explanation

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

### **NEW QUESTION: 60**

Which of the following state management command allow you to retrieve a list of resources that are part of the state file?

- A. terraform state list
- B. terraform state view
- C. terraform view
- D. terraform list

**Answer: A ([LEAVE A REPLY](#))**

Explanation

The terraform state list command is used to list resources within a Terraform state.

Usage: terraform state list [options] [address...]

The command will list all resources in the state file matching the given addresses (if any). If no addresses are given, all resources are listed.

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

## NEW QUESTION: 61

State locking does not happen automatically and must be specified at run

- A. False
- B. True

**Answer: A ([LEAVE A REPLY](#))**

Explanation

State locking happens automatically on all operations that could write state.

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

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**

**Valid TA-002-P Dumps** shared by Fast2test.com for Helping Passing TA-002-P Exam!

Fast2test.com now offer the **newest TA-002-P exam dumps**, the Fast2test.com

TA-002-P exam **questions have been updated** and **answers have been corrected**

get the **newest** Fast2test.com TA-002-P dumps with Test Engine here:

<https://www.fast2test.com/TA-002-P-premium-file.html> (324 Q&As Dumps, **30%OFF**

**Special Discount: freecram)**