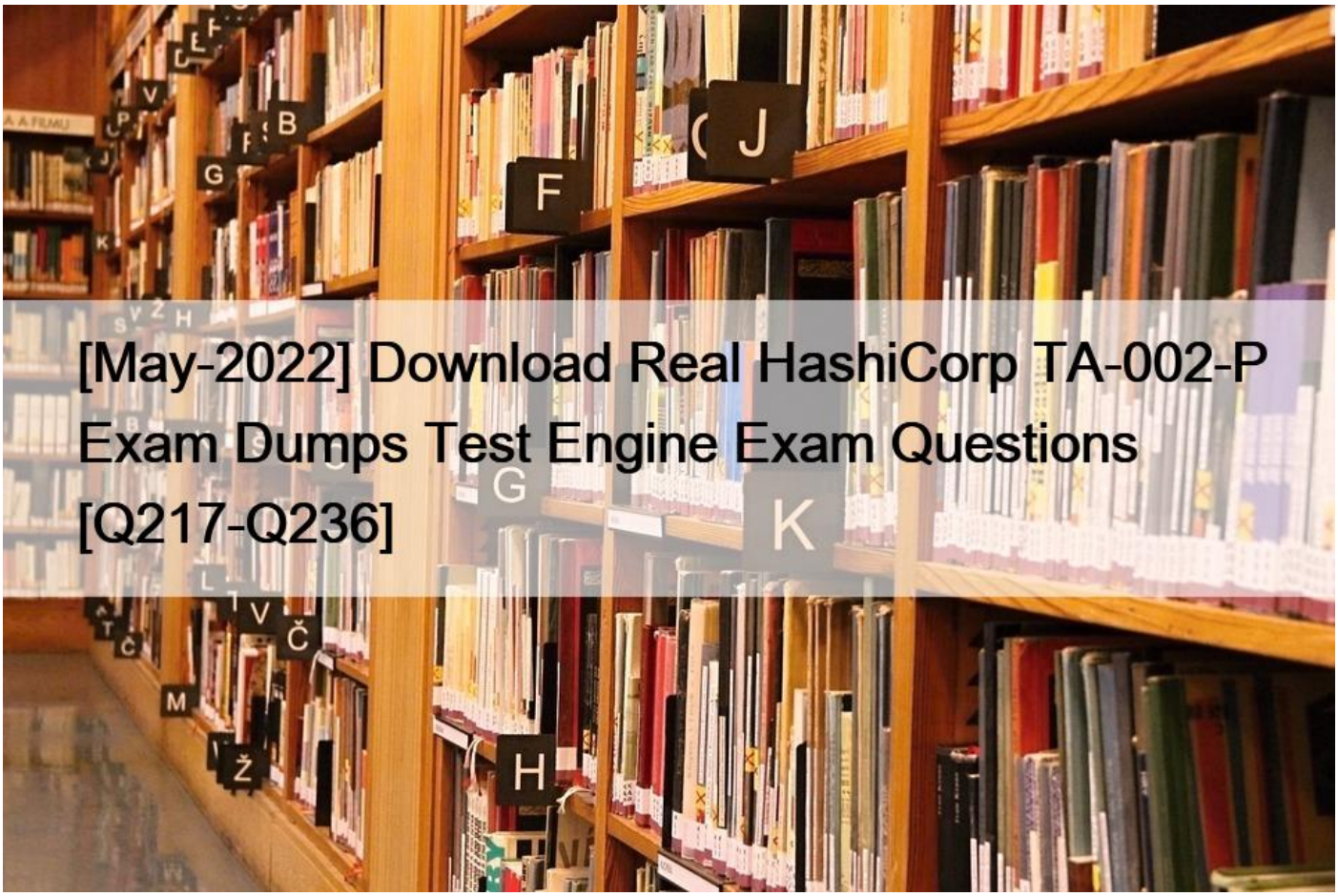
[May-2022 Download Real HashiCorp TA-002-P Exam Dumps Test Engine Exam Questions [Q217-Q236



[May-2022] Download Real HashiCorp TA-002-P Exam Dumps Test Engine Exam Questions
New TA-002-P exam dumps Use Updated HashiCorp Exam

NEW QUESTION 217

By default, provisioners that fail will also cause the Terraform apply itself to error. How can you change this default behavior within a provisioner?
* provisioner “local-exec” { on_failure = “next” }
* provisioner “local-exec” { when = “failure” terraform apply }
* provisioner “local-exec” { on_failure = “continue” }
* provisioner “local-exec” { on_failure = continue }
https://www.terraform.io/docs/provisioners/index.html

NEW QUESTION 218

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?
* Terraform attempts to provision the resource up to three times before exiting with an error

* the terraform plan is rolled back and all provisioned resources are removed
* it is automatically deleted
* the resource is marked as tainted

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 219

After executing a terraform apply, you notice that a resource has a tilde (~) next to it. What does this infer?

* The resource will be updated in place.
* The resource will be created.
* Terraform can’t determine how to proceed due to a problem with the state file.
* The resource will be destroyed and recreated.

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.

NEW QUESTION 220

What information does the public Terraform Module Registry automatically expose about published modules?
* Required input variables
* Optional inputs variables and default values
* Outputs
* All of the above
* None of the above
Reference: https://www.terraform.io/docs/registry/modules/publish.html

NEW QUESTION 221

Only the user that generated a plan may apply it.
* True
* False
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 222

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?
* Terraform registry
* custom VCS integration
* private module registry
* remote runs

NEW QUESTION 223

What features stops multiple admins from changing the Terraform state at the same time?
* Version control
* Backend types
* Provider constraints
* State locking

NEW QUESTION 224

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

Explanation

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

NEW QUESTION 225

I78

correct?
* When using local state, the state file is stored in plain-text.
* The state file is always encrypted at rest.
* Storing state remotely can provide better security.
* Using the mask feature, you can instruct Terraform to mask sensitive data in the state file.
* The Terraform state can contain sensitive data, therefore the state file should be protected from unauthorized access.
* Terraform Cloud always encrypts state at rest.

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 226

During a terraform plan, a resource is successfully created but eventually fails during provisioning. What happens to the resource?
* Terraform attempts to provision the resource up to three times before exiting with an error
* the terraform plan is rolled back and all provisioned resources are removed
* it is automatically deleted
* 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.

NEW QUESTION 227

A Terraform provisioner must be nested inside a resource configuration block.
* True
* False
Explanation

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

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

NEW QUESTION 228

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?
* False
* True

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 229

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)
* The module must be PCI/HIPPA compliant.
* Module repositories must use this three-part name format, terraform– .
* The registry uses tags to identify module versions.
* Release tag names must be for the format x.y.z, and can optionally be prefixed with a v .
* The module must be on GitHub and must be a public repo.
https://www.terraform.io/docs/registry/modules/publish.html#requirements

NEW QUESTION 230

terraform apply will fail if you have not run terraform plan first to update the plan output.
* True
* False

NEW QUESTION 231

Workspaces in Terraform provides similar functionality in the open-source, Terraform Cloud, and Enterprise versions of Terraform.
* True
* False
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 232

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. }
* The regular expression named module.web_server
* The output of a module named web_server
* By querying the AWS EC2 API to retrieve the IP address
* Value of the web_server parameter from the variables.tf file

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 233

Terraform works well in Windows but a Windows server is required.
* False
* True

You may see this

Terraform does not require GO language to be installed as a prerequisite and it does not require a Windows Server as well.

NEW QUESTION 234

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?
* Terraform will remove the VM from state file
* Terraform will report an error
* Terraform will not make any changes
* Terraform will recreate the VM

NEW QUESTION 235

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?
* var.images[“us-west-1”]
* var.images[3]
* var.images[2]
* lookup(var.images[“us-west-1”]

NEW QUESTION 236

Provider dependencies are created in several different ways. Select the valid provider dependencies from the following list: (select three)
* Explicit use of a provider block in configuration, optionally including a version constraint.
* Use of any resource belonging to a particular provider in a resource or data block in configuration.
* Existence of any resource instance belonging to a particular provider in the current state.
* 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