[Q75-Q98 Certification Training for VA-002-P Exam Dumps Test Engine [2024

Certification Training for VA-002-P Exam Dumps Test Engine [2024]
Nov 12, 2024 Step by Step Guide to Prepare for VA-002-P Exam

HashiCorp VA-002-P certification exam is designed to validate one's skills and knowledge in using HashiCorp Vault as a secrets management tool. VA-002-P exam is intended for professionals who have experience in IT operations, DevOps, or security engineering and who are interested in learning more about Vault.

NO.75 What Terraform feature is shown in the example below?

1. resource “aws_security_group” “example” {

2. name = “sg-app-web-01”

3. dynamic “ingress” {

4. for_each = var.service_ports

5. content {

6. from_port = ingress.value

7. to_port = ingress.value

8. protocol = “tcp”

9. }

10. }

11. }
* data source
* dynamic block
* local values
* conditional expression

You can dynamically construct repeatable nested blocks like ingress using a special dynamic block type, which is supported inside resource, data, provider, and provisioner blocks

NO.76 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 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.
* Module repositories must use this three-part name format, terraform-<PROVIDER>-<NAME>.
* The module must be PCI/HIPPA compliant.
* The module must be on GitHub and must be a public repo

The list below contains all the requirements for publishing a module. Meeting the requirements for publishing a module is extremely easy. The list may appear long only to ensure we’re detailed, but adhering to the requirements should happen naturally.

GitHub. The module must be on GitHub and must be a public repo. This is only a requirement for the public registry. If you’re using a private registry, you may ignore this requirement.

Named terraform-<PROVIDER>-<NAME>. Module repositories must use this three-part name format, where <NAME> reflects the type of infrastructure the module manages, and <PROVIDER> is the main provider where it creates that infrastructure. The <NAME> segment can contain additional hyphens. Examples: terraform-google-vault or terraform-aws-ec2-instance.

Repository description. The GitHub repository description is used to populate the short description of the module. This should be a simple one-sentence description of the module.

Standard module structure. The module must adhere to the standard module structure. This allows the registry to inspect your module and generate documentation, track resource usage, parse submodules and examples, and more.

x.y.z tags for releases. The registry uses tags to identify module versions. Release tag names must be a semantic version, which can optionally be prefixed with a v. For example, v1.0.4 and 0.9.2. To publish a module initially, at least one release tag must be present. Tags that don’t look like version numbers are ignored.

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

NO.77 How can Vault be used to programmatically obtain a generated code for MFA, somewhat similar to Google Authenticator?
* cubbyhole
* the identity secrets engine
* TOTP secrets engine
* the random byte generator

The TOTP secrets engine generates time-based credentials according to the TOTP standard. The secrets engine can also be used to generate a new key and validate passwords generated by that key.

The TOTP secrets engine can act as both a generator (like Google Authenticator) and a provider (like the Google.com sign-in service).

As a Generator

The TOTP secrets engine can act as a TOTP code generator. In this mode, it can replace traditional TOTP generators like Google Authenticator. It provides an added layer of security since the ability to generate codes is guarded by policies and the entire process is audited.

Reference link:- https://www.vaultproject.io/docs/secrets/totp

NO.78 In order to make a Terraform configuration file dynamic and/or reusable, static values should be converted to use what?
* regular expressions
* module
* input parameters
* output value

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.79 You’ve decided to use AWS KMS to automatically unseal Vault on private EC2 instances. After deploying your Vault cluster, and running vault operator init, Vault responds with an error and cannot be unsealed.

You’ve determined that the subnet you’ve deployed Vault into doesn’t have internet access. What can you do to enable Vault to communicate with AWS KMS in the most secure way?
* ask the networking team to provide Vault with inbound access from the internet
* deploy Vault in a public subnet and provide the Vault nodes with public IP addresses
* add a VPC endpoint
* change the permissions on the Internet Gateway to allow the Vault nodes to communicate over the Internet

In this particular question, a VPC endpoint can provide private connectivity to an AWS service without having to traverse the public internet. This way you hit a private endpoint for the service rather than connecting to the public endpoint.

This is more of an AWS-type question, but the underlying premise still holds regardless of where your Vault cluster is deployed. If you use a public cloud KMS solution, such as AWS KMS, Azure Key Vault, GCP Cloud KMS, or AliCloud KMS, your Vault cluster will need the ability to communicate with that service to unseal itself.

NO.80 When configuring Vault replication and monitoring its status, you keep seeing something called ‘WALs’. What are WALs?
* wake after lan
* warning of allocated logs
* write-ahead log
* write along logging

Reference links:-

https://learn.hashicorp.com/vault/day-one/monitor-replication

https://www.vaultproject.io/docs/internals/replication

NO.81 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.
* TF_VAR
* TF_VAR_NAME
* TF_ENV
* TF_ENV_VAR
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

NO.82 What are some of the problems of how infrastructure was traditionally managed before Infrastructure as Code? (select three)
* Requests for infrastructure or hardware required a ticket, increasing the time required to deploy applications
* 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
* Traditionally managed infrastructure can’t keep up with cyclic or elastic applications
* 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
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 change of human error.

NO.83 True or False: You can migrate the Terraform backend but only if there are no resources currently being managed.
* False
* True
If you are already using Terraform to manage infrastructure, you probably want to transfer 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.

NO.84 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 private install
* disconnected

* non-traditional
* air-gapped

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.

NO.85 By default, the max TTL for a token is how many days?
* 14 days
* 32 days
* 31 days
* 7 days

The system max TTL, which is 32 days but can be changed in Vault’s configuration file.

The max TTL set on a mount using mount tuning. This value is allowed to override the system max TTL — it can be longer or shorter, and if set this value will be respected.

A value suggested by the auth method that issued the token. This might be configured on a per-role, per-group, or per-user basis. This value is allowed to be less than the mount max TTL (or, if not set, the system max TTL), but it is not allowed to be longer.

Reference link:- https://www.vaultproject.io/docs/concepts/tokens

NO.86 During a terraform apply, a resource is successfully created but eventually fails during provisioning. What happens to the resource?
* Terraform attempts to provide 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.

NO.87 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)
* CVS Version Control
* GitHub Enterprise
* Bitbucket Cloud
* Azure DevOps Server
* GitHub

Terraform Cloud supports the following VCS providers:

– GitHub

– GitHub.com (OAuth)

– GitHub Enterprise

– GitLab.com

– GitLab EE and CE

– Bitbucket Cloud

– Bitbucket Server

– Azure DevOps Server

– Azure DevOps Services

https://www.terraform.io/docs/cloud/vcs/index.html#supported-vcs-providers

NO.88 What type of policy is shown below?

1. key_prefix “vault/” {

2. policy = “write”

3. }

4. node_prefix “” {

5. policy = “write”

6. }

7. service “vault” {

8. policy = “write”

9. }

10. agent_prefix “” {

11. policy = “write”

12. }

13. session_prefix “” {

14. policy = “write”

15. }
* Vault policy allowing access to certain paths
* Consul ACL policy for a Vault node
* Consul configuration policy to enable Consul features
* Vault token policy is written for a user

If using ACLs in Consul, you’ll need appropriate permissions. For Consul 0.8, these policies will work for most use-cases,

assuming that your service name is vault and the prefix being used is vault/Consul ACLs should always be enabled when using Consul as a storage backend. This policy allows Vault to communicate to the required services hosted on Consul.

Reference link:- https://www.vaultproject.io/docs/configuration/storage/consul

NO.89 In Terraform Enterprise, a workspace can be mapped to how many VCS repos?
* 5
* 3
* 2
* 1
A workspace can only be configured to a single VCS repo, however, multiple workspaces can use the same repo, if needed. A good Explanation: of how to configure your code repositories can be found here.

NO.90 Which auth method is ideal for machine to machine authentication?
* GitHub
* UserPass
* AppRole
* Okta
The ideal method for a machine to machine authentication is AppRole although it’s not the only method. The other options are frequently reserved for human access.

Reference link:- https://www.hashicorp.com/blog/authenticating-applications-with-vault-approle/

NO.91 Select the answer below that completes the following statement:

Terraform Cloud can be managed from the CLI but requires _____?
* a TOTP token
* a username and password
* authentication using MFA
* an API token
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.92 Select the operating systems which are supported for a clustered Terraform Enterprise: (select four)
* Unix
* Amazon Linux
* Red Hat
* Ubuntu
* CentOS
Note: (5/27/20) This Question: has been recently updated to reflect documentation updates on the HashiCorp website. It seems they have removed the clustering-specific requirements and are now following the standard Enterprise operating system requirements.

Terraform Enterprise currently supports running under the following operating systems for a Clustered deployment:

– Ubuntu 16.04.3 – 16.04.5 / 18.04

– Red Hat Enterprise Linux 7.4 through 7.7

– CentOS 7.4 – 7.7

– Amazon Linux

– Oracle Linux

Clusters currently don’t support other Linux variants.

https://www.terraform.io/docs/enterprise/before-installing/index.html#operating-system-requirements

NO.93 From the code below, identify the implicit dependency:

1. resource “aws_eip” “public_ip” {

2. vpc = true

3. instance = aws_instance.web_server.id

4. }

5. resource “aws_instance” “web_server” {

6. ami = “ami-2757f631”

7. instance_type = “t2.micro”

8. depends_on = [aws_s3_bucket.company_data]

9. }
* The EC2 instance labeled web_server
* The EIP with an id of ami-2757f631
* The AMI used for the EC2 instance
* The S3 bucket labeled company_data

The EC2 instance labeled web_server is the implicit dependency as the aws_eip cannot be created until the aws_instance labeled web_server has been provisioned and the id is available.

Note that aws_s3_bucket.example is an explicit dependency.

NO.94 In terraform, most resource dependencies are handled automatically. Which of the following statements describes best how terraform resource dependencies are handled?
* 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 much be running the latest version of Terraform.
* 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.
* 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.
* Resource dependencies are handled automatically by the depends_on meta_argument, which is set to true by default.

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.

https://www.terraform.io/docs/configuration/resources.html

NO.95 What happens when a terraform plan is executed?
* the backend is initialized and the working directory is prepped
* creates an execution plan and determines what changes are required to achieve the desired state in the configuration files.
* applies the changes required in the target infrastructure in order to reach the desired configuration
* reconciles the state Terraform knows about with the real-world infrastructure

The terraform plan command is used to create an execution plan. Terraform performs a refresh, unless explicitly disabled, and then determines what actions are necessary to achieve the desired state specified in the configuration files.

After a plan has been run, it can be executed by running a terraform apply

NO.96 The userpass auth method has the ability to access external services in order to provide authentication to Vault.
* FALSE
* TRUE

The userpass auth method uses a local database that cannot interact with any services outside of the Vault instance.

NO.97 What does the command terraform fmt do?
* formats the state file in order to ensure the latest state of resources can be obtained
* updates the font of the configuration file to the official font supported by HashiCorp
* rewrite Terraform configuration files to a canonical format and style
* deletes the existing configuration file

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.

NO.98 By default, where does Terraform store its state file?
* shared directory
* current working directory
* Amazon S3 bucket
* remotely using Terraform Cloud

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.

HashiCorp VA-002-P certification exam is a valuable certification for IT professionals who want to specialize in the use of HashiCorp's Vault product. It validates the candidate's knowledge and skills in the use of Vault, which is a critical component of any organization's security strategy. With the increasing demand for Vault expertise, obtaining this certification can help professionals

stay relevant and competitive in the industry.

Ultimate Guide to Prepare VA-002-P Certification Exam for HashiCorp Security Automation:
https://www.dumpsmaterials.com/VA-002-P-real-torrent.html]