VAULT-ASSOCIATE$^{Q\&As}$

HashiCorp Certified: Vault Associate (002)

Pass HashiCorp VAULT-ASSOCIATE Exam with 100% Guarantee

QUESTION 1

What can be used to limit the scope of a credential breach?

A. Storage of secrets in a distributed ledger

B. Enable audit logging

C. Use of a short-lived dynamic secrets

D. Sharing credentials between applications

Correct Answer: C

Using a short-lived dynamic secrets can help limit the scope of a credential breach by reducing the exposure time of the secrets. Dynamic secrets are generated on- demand by Vault and automatically revoked when they are no longer needed. This way, the credentials are not stored in plain text or in a static database, and they can be rotated frequently to prevent unauthorized access. Dynamic secrets also provide encryption as a service, which means that they perform cryptographic operations on data in-transit without storing any data. This adds an extra layer of security and reduces the risk of data leakage or tampering. References: Dynamic secrets | Vault | HashiCorp Developer, What are dynamic secrets and why do I need them? - HashiCorp

QUESTION 2

Where can you set the Vault seal configuration? Choose two correct answers.

A. Cloud Provider KMS

B. Vault CLI

C. Vault configuration file

D. Environment variables

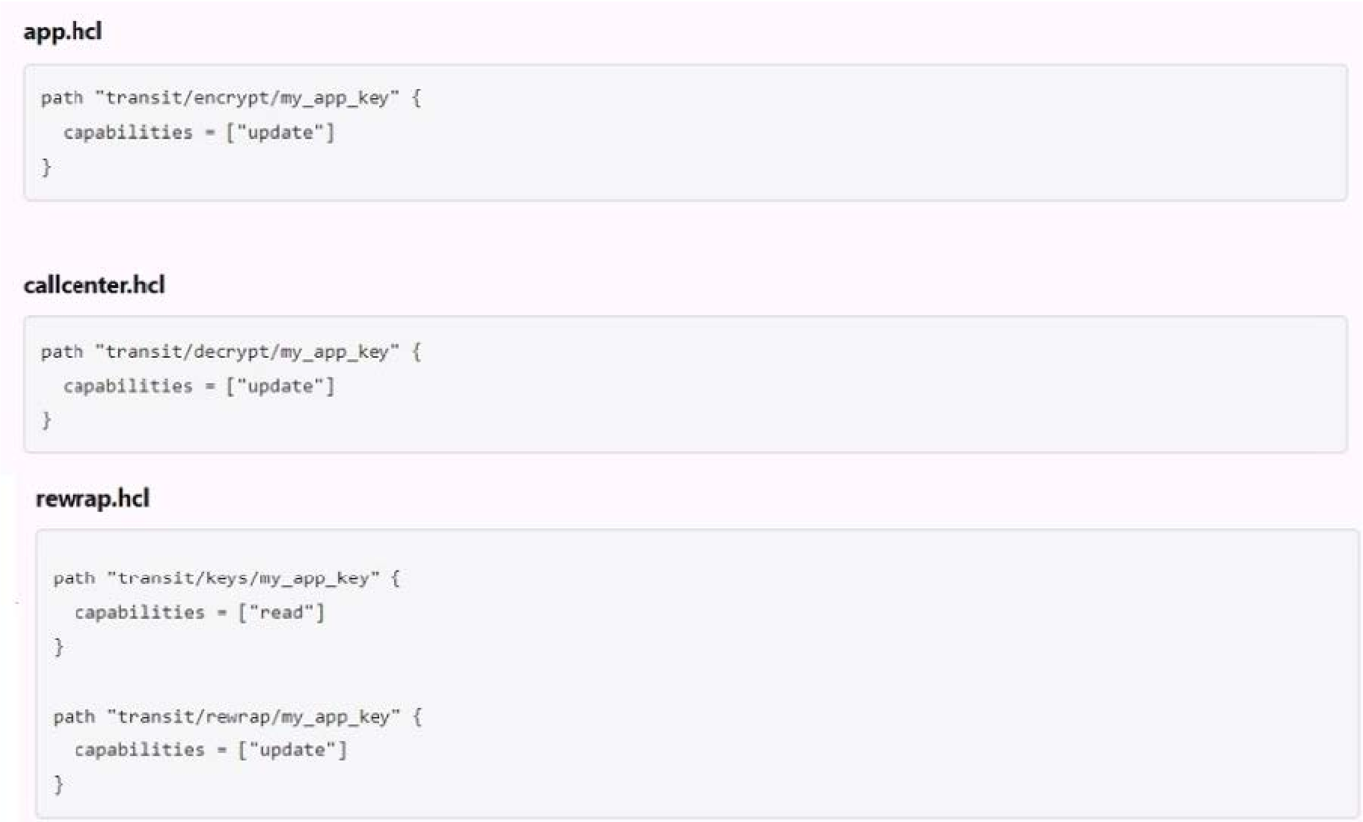
E. Vault API

Correct Answer: CD

The Vault seal configuration can be set in two ways: through the Vault configuration file or through environment variables. The Vault configuration file is a text file that contains the settings and options for Vault, such as the storage backend, the listener, the telemetry, and the seal. The seal stanza in the configuration file specifies the seal type and the parameters to use for additional data protection, such as using HSM or Cloud KMS solutions to encrypt and decrypt the root key. The seal configuration can also be set through environment variables, which will take precedence over the values in the configuration file. The environment variables are prefixed with VAULT_SEAL_ and followed by the seal type and the parameter name. For example, VAULT_SEAL_AWSKMS_REGION sets the region for the AWS KMS seal. References: Seals - Configuration | Vault | HashiCorp Developer, Environment Variables | Vault | HashiCorp Developer

QUESTION 3

The following three policies exist in Vault. What do these policies allow an organization to do?



app.hcl

```
path "transit/encrypt/my_app_key" {
  capabilities = ["update"]
}
```

callcenter.hcl

```
path "transit/decrypt/my_app_key" {
  capabilities = ["update"]
}
```

rewrap.hcl

```
path "transit/keys/my_app_key" {
  capabilities = ["read"]
}

path "transit/rewrap/my_app_key" {
  capabilities = ["update"]
}
```

A. Separates permissions allowed on actions associated with the transit secret engine

B. Nothing, as the minimum permissions to perform useful tasks are not present

C. Encrypt, decrypt, and rewrap data using the transit engine all in one policy

D. Create a transit encryption key for encrypting, decrypting, and rewrapping encrypted data

Correct Answer: C

The three policies that exist in Vault are: admins: This policy grants full access to all secrets and operations in Vault. It can be used by administrators or operators who need to manage all aspects of Vault. default: This policy grants access to all secrets and operations in Vault except for those that require specific policies. It can be used as a fallback policy when no other policy matches. transit: This policy grants access only to the transit secrets engine, which handles cryptographic functions on data in-transit. It can be used by applications or services that need to encrypt or decrypt data using Vault. These policies allow an organization to perform useful tasks such as: Encrypting, decrypting, and rewrapping data using the transit engine all in one policy: This policy grants access to both the transit secrets engine and the default policy, which allows performing any operation on any secret in Vault. Creating a transit encryption key for encrypting, decrypting, and rewrapping encrypted data: This policy grants access only to the transit secrets engine and its associated keys, which are used for encrypting and decrypting data in transit using AES-GCM with a 256-bit AES key or other supported key types. Separating permissions allowed on actions associated with the transit secret engine: This policy grants access only to specific actions related to the transit secrets engine, such as creating keys or wrapping requests. It does not grant access to other operations or secrets in Vault.

QUESTION 4

Your DevOps team would like to provision VMs in GCP via a CICD pipeline. They would like to integrate Vault to protect the credentials used by the tool. Which secrets engine would you recommend?

A. Google Cloud Secrets Engine

B. Identity secrets engine

C. Key/Value secrets engine version 2

D. SSH secrets engine

Correct Answer: A

The Google Cloud Secrets Engine is the best option for the DevOps team to provision VMs in GCP via a CICD pipeline and integrate Vault to protect the credentials used by the tool. The Google Cloud Secrets Engine can dynamically generate GCP service account keys or OAuth tokens based on IAM policies, which can be used to authenticate and authorize the CICD tool to access GCP resources. The credentials are automatically revoked when they are no longer used or when the lease expires, ensuring that the credentials are short-lived and secure. The DevOps team can configure rolesets or static accounts in Vault to define the scope and permissions of the credentials, and use the Vault API or CLI to request credentials on demand. The Google Cloud Secrets Engine also supports generating access tokens for impersonated service accounts,which can be useful for delegating access to other service accounts without storing or managing their keys1. The Identity Secrets Engine is not a good option for this use case, because it does not generate GCP credentials, but rather generates identity tokens that can be used to access other Vault secrets engines or namespaces2. The Key/Value Secrets Engine version 2 is also not a good option, because it does not generate dynamic credentials, but rather stores and manages static secrets that the user provides3. The SSH Secrets Engine is not a good option either, because it does not generate GCP credentials, but rather generates SSH keys or OTPs that can be used to access remote hosts via SSH4. References: Google Cloud - Secrets Engines | Vault | HashiCorp Developer Identity - Secrets Engines | Vault | HashiCorp Developer KV - Secrets Engines | Vault | HashiCorp Developer SSH - Secrets Engines | Vault | HashiCorp Developer

QUESTION 5

Where does the Vault Agent store its cache?

A. In a file encrypted using the Vault transit secret engine

B. In the Vault key/value store

C. In an unencrypted file

D. In memory

Correct Answer: D

The Vault Agent stores its cache in memory, which means that it does not persist the cached tokens and secrets to disk or any other storage backend. This makes the cache more secure and performant, as it avoids exposing the sensitive data to potential attackers or unauthorized access. However, this also means that the cache is volatile and will be lost if the agent process is terminated or restarted. To mitigate this, the agent can optionally use a persistent cache file to restore the tokens and leases from a previous agent process. The persistent cache file is encrypted using a key derived from the agent\\'s auto-auth token and a nonce, and it is stored in a user-specified location on disk. References: Caching Vault Agent | Vault | HashiCorp Developer, Vault Agent Persistent Caching | Vault | HashiCorp Developer