Free Questions for Vault-Associate by dumpssheet

Shared by Hendrix on 29-01-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

An authentication method should be selected for a use case based on:

Options:

A- The auth method that best establishes the identity of the client

B- The cloud provider for which the client is located on

C- The strongest available cryptographic hash for the use case

D- Compatibility with the secret engine which is to be used

Answer:

A

Explanation:

An authentication method should be selected for a use case based on the auth method that best establishes the identity of the client. The identity of the client is the basis for assigning a set of policies and permissions to the client in Vault. Different auth methods have different ways of verifying the identity of the client, such as using passwords, tokens, certificates, cloud credentials, etc. Depending on

the use case, some auth methods may be more suitable or convenient than others. For example, for human users, the userpass or ldap auth methods may be easy to use, while for machines or applications, the approle or aws auth methods may be more secure and scalable. The choice of the auth method should also consider the trade-offs between security, performance, and usability.Reference:Auth Methods | Vault | HashiCorp Developer,Authentication - Concepts | Vault | HashiCorp Developer

Question 2

Question Type: MultipleChoice

To make an authenticated request via the Vault HTTP API, which header would you use?

Options:

A- The X-Vault-Token HTTP Header

B- The x-Vault-Request HTTP Header

C- The Content-Type HTTP Header

D- The X-Vault-Namespace HTTP Header

Answer:

A

Explanation:

To make an authenticated request via the Vault HTTP API, you need to use the X-Vault-Token HTTP Header or the Authorization HTTP Header using the Bearer <token> scheme. The token is a string that represents your identity and permissions in Vault. You can obtain a token by using an authentication method, such as userpass, approle, aws, etc. The token can also be a root token, which has unlimited access to Vault, or a wrapped token, which is a response-wrapping token that can be used to unwrap the actual token. The token must be sent with every request to Vault that requires authentication, except for the unauthenticated endpoints, such as sys/init, sys/seal-status, sys/unseal, etc.The token is used by Vault to verify your identity and enforce the policies that grant or deny access to various paths and operations.Reference: https://developer.hashicorp.com/vault/api-docs3, https://developer.hashicorp.com/vault/docs/concepts/tokens4, https://developer.hashicorp.com/vault/docs/concepts/auth5

Question 3

Question Type: MultipleChoice

What does the following policy do?

```
path "secret/data/{{identity.entity.id}}/*" {
  capabilities = ["create", "update", "read", "delete"]
}

path "secret/metadata/{{identity.entity.id}}/*" {
  capabilities = ["list"]
}
```

Options:

A- Grants access for each user to a KV folder which shares their id

B- Grants access to a special system entity folder

C- Allows a user to read data about the secret endpoint identity

D- Nothing, this is not a valid policy

Answer:

C

Explanation:

This policy allows a user to read data about the secret endpoint identity. The policy grants the user the ability to create, update, read, and delete data in the "secret/data/{identity.entity.id}" path. Additionally, the user is allowed to list data in the "secret/metadata/{identity.entity.id}" path. This policy is useful for users who need to access information about the secret endpoint identity.

The secret endpoint identity is a feature of the Identity Secrets Engine, which allows Vault to generate identity tokens that can be used to access other Vault secrets engines or namespaces. The identity tokens are based on the entity and group information of the user or machine that authenticates with Vault. The entity is a unique identifier for the user or machine, and the group is a collection of entities that share some common attributes. The identity tokens can carry metadata and policies that are associated with the entity and group.

The "secret/data/{identity.entity.id}" path is where the user can store and retrieve data that is related to the secret endpoint identity. For example, the user can store some configuration or preferences for the secret endpoint identity in this path. The "secret/metadata/{identity.entity.id}" path is where the user can list the metadata of the data stored in the "secret/data/{identity.entity.id}" path. For example, the user can list the version, creation time, deletion time, and destroy time of the data in this path.

[Identity - Secrets Engines | Vault | HashiCorp Developer]

[KV - Secrets Engines | Vault | HashiCorp Developer]

Question 4

Question Type: **MultipleChoice**

You are using Vault's Transit secrets engine to encrypt your dat

a. You want to reduce the amount of content encrypted with a single key in case the key gets compromised. How would you do this?

Options:

A- Use 4096-bit RSA key to encrypt the data

B- Upgrade to Vault Enterprise and integrate with HSM

C- Periodically re-key the Vault's unseal keys

D- Periodically rotate the encryption key

Answer:

D

Explanation:

The Transit secrets engine supports the rotation of encryption keys, which allows you to change the key that is used to encrypt new data without affecting the ability to decrypt data that was already encrypted. This reduces the amount of content encrypted with a single key in case the key gets compromised, and also helps you comply with the NIST guidelines for key rotation. You can rotate the encryption key manually by invoking the /transit/keys/<name>/rotate endpoint, or you can configure the key to automatically rotate based on a time interval or a number of encryption operations. When you rotate a key, Vault generates a new key version and increments the key's latest_version metadata. The new key version becomes the encryption key used for encrypting any new data. The previous key versions

are still available for decrypting the existing data, unless you specify a minimum decryption version to archive the old key versions.You can also delete or disable old key versions if you want to revoke access to the data encrypted with those versions.Reference: https://developer.hashicorp.com/vault/docs/secrets/transit1, https://developer.hashicorp.com/vault/api-docs/secret/transit2

Question 5

Question Type: MultipleChoice

An organization wants to authenticate an AWS EC2 virtual machine with Vault to access a dynamic database secret. The only authentication method which they can use in this case is AWS.

Options:

A- True

B- False

Answer:

B

Explanation:

The statement is false. An organization can authenticate an AWS EC2 virtual machine with Vault to access a dynamic database secret using more than one authentication method. The AWS auth method is one of the options, but not the only one. The AWS auth method supports two types of authentication: ec2 and iam. The ec2 type uses the signed EC2 instance identity document to authenticate the EC2 instance. The iam type uses the AWS Signature v4 algorithm to sign a request to the sts:GetCallerIdentity API and authenticate the IAM principal. However, the organization can also use other auth methods that are compatible with EC2 instances, such as AppRole, JWT/OIDC, or Kubernetes. These methods require the EC2 instance to have some sort of identity material, such as a role ID, a secret ID, a JWT token, or a service account token, that can be used to authenticate to Vault. The identity material can be provisioned to the EC2 instance using various mechanisms, such as user data, metadata service, or cloud-init scripts. The choice of the auth method depends on the use case, the security requirements, and the trade-offs between convenience and control.Reference:AWS - Auth Methods | Vault | HashiCorp Developer,AppRole - Auth Methods | Vault | HashiCorp Developer,JWT/OIDC - Auth Methods | Vault | HashiCorp Developer,Kubernetes - Auth Methods | Vault | HashiCorp Developer

Question 6

Question Type: MultipleChoice

Which of the following vault lease operations uses a lease _ id as an argument? Choose two correct answers.

Options:

A- renew

B- revoke -prefix

C- create

D- describe

E- revoke

Answer:

A, E

Explanation:

The vault lease operations that use a lease_id as an argument are renew and revoke. The renew operation allows a client to extend the validity of a lease associated with a secret or a token. The revoke operation allows a client to terminate a lease immediately and invalidate the secret or the token. Both operations require a lease_id as an argument to identify the lease to be renewed or revoked. The lease_id can be obtained from the response of reading a secret or creating a token, or from the vault lease list command. The other operations, revoke-prefix, create, and describe, do not use a lease_id as an argument. The revoke-prefix operation allows a client to revoke all secrets or tokens generated under a given prefix. The create operation allows a client to create a new lease for a secret. The describe operation allows a client to view information about a lease, such as its TTL, policies, and metadata.Reference:Lease, Renew, and Revoke | Vault | HashiCorp Developer,vault lease - Command | Vault | HashiCorp Developer

Question 7

Question Type: MultipleChoice

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

Options:

A- Cloud Provider KMS

B- Vault CLI

C- Vault configuration file

D- Environment variables

E- Vault API

Answer:

C, D

Explanation:

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.Reference:Seals - Configuration | Vault | HashiCorp Developer,Environment Variables | Vault | HashiCorp Developer

Question 8

Question Type: **MultipleChoice**

When creating a policy, an error was thrown:

‹ ACL Policies

Create ACL policy

Error

failed to parse policy: path "secret/webapp/*": invalid capability "write"

Name

webapp

Policy Upload file

```
1 path "secret/webapp/*" {
2   capabilities = ["read", "write", "delete", "list", "sudo"]
3 }
```

Which statement describes the fix for this issue?

Options:

A- Replace write with create in the capabilities list

B- You cannot have a wildcard (' * ') in the path

C- sudo is not a capability

Answer:

A

Explanation:

The error was thrown because the policy code contains an invalid capability, "write". The valid capabilities for a policy are "create", "read", "update", "delete", "list", and "sudo". The "write" capability is not recognized by Vault and should be replaced with "create", which allows creating new secrets or overwriting existing ones. The other statements are not correct, because the wildcard (*) and the sudo capability are both valid in a policy. The wildcard matches any number of characters within a path segment, and the sudo capability allows performing certain operations that require root privileges.

[Policy Syntax | Vault | HashiCorp Developer]

[Policy Syntax | Vault | HashiCorp Developer]

Question 9

Question Type: MultipleChoice

As a best practice, the root token should be stored in which of the following ways?

Options:

A- Should be revoked and never stored after initial setup

B- Should be stored in configuration automation tooling

C- Should be stored in another password safe

D- Should be stored in Vault

Answer:

A

Explanation:

The root token is the initial token created when initializing Vault. It has unlimited privileges and can perform any operation in Vault. As a best practice, the root token should be revoked and never stored after initial setup. This is because the root token is a single point of failure and a potential security risk if it is compromised or leaked. Instead of using the root token, Vault operators should create other tokens with appropriate policies and roles that allow them to perform their tasks. If a new root token is needed in an emergency, the vault operator generate-root command can be used to create one on-the-fly with the consent of a quorum of unseal key holders.Reference:Tokens | Vault | HashiCorp Developer,Generate root tokens using unseal keys | Vault | HashiCorp Developer

Question 10

Question Type: MultipleChoice

When unsealing Vault, each Shamir unseal key should be entered:

Options:

A- Sequentially from one system that all of the administrators are in front of

B- By different administrators each connecting from different computers

B- While encrypted with each administrators PGP key

D- At the command line in one single command

Answer:

B, B

Explanation:

When unsealing Vault, each Shamir unseal key should be entered by different administrators each connecting from different computers. This is because the Shamir unseal keys are split into shares that are distributed to trusted operators, and no single operator should have access to more than one share. This way, the unseal process requires the cooperation of a quorum of key holders, and enhances the security and availability of Vault. The unseal keys can be entered via multiple mechanisms from multiple client machines, and the process is stateful. The order of the keys does not matter, as long as the threshold number of keys is reached. The unseal keys should not be entered at the command line in one single command, as this would expose them to the history and compromise the security.The unseal keys should not be encrypted with each administrator's PGP key, as this would prevent Vault from decrypting them and reconstructing the master key.Reference: https://developer.hashicorp.com/vault/docs/concepts/seal3, https://developer.hashicorp.com/vault/docs/commands/operator/unseal

Question 11

Question Type: MultipleChoice

Your organization has an initiative to reduce and ultimately remove the use of long lived X.509 certificates. Which secrets engine will best support this use case?

Options:

A- PKI

B- Key/Value secrets engine version 2, with TTL defined

C- Cloud KMS

D- Transit

Answer:

A

Explanation:

The PKI secrets engine is designed to support the use case of reducing and ultimately removing the use of long lived X.509 certificates. The PKI secrets engine can generate dynamic X.509 certificates on demand, with short time-to-live (TTL) and automatic revocation. This eliminates the need for manual processes of generating, signing, and rotating certificates, and reduces the risk of certificate compromise or misuse. The PKI secrets engine can also act as a certificate authority (CA) or an intermediate CA, and can integrate with external CAs or CRLs.The PKI secrets engine can issue certificates for various purposes, such as TLS, SSH, code signing, email encryption, etc.Reference: https://developer.hashicorp.com/vault/docs/secrets/pki1, https://developer.hashicorp.com/vault/tutorials/getting-started/getting-started-dynamic-secrets

Question 12

Question Type: MultipleChoice

Where does the Vault Agent store its cache?

Options:

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

Answer:

D

Explanation:

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.Reference:Caching - Vault Agent | Vault | HashiCorp Developer,Vault Agent Persistent Caching | Vault | HashiCorp Developer