Free Questions for Vault-Associate by go4braindumps

Shared by Roth on 24-05-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

To give a role the ability to display or output all of the end points under the /secrets/apps/* end point it would need to have which capability set?

Options:

A- update

B- read

C- sudo

D- list

E- None of the above

Answer:

C

Explanation:

To give a role the ability to display or output all of the end points under the /secrets/apps/* end point, it would need to have the list capability set. The list capability allows a role to perform any operation on any path in Vault, including reading, writing, deleting, and listing. The list capability is required for roles that need to access sensitive data or perform administrative tasks in Vault. The other capabilities are not relevant for this scenario, as they only allow specific operations on specific paths or secrets engines.Reference:Policies | Vault | HashiCorp Developer,token capabilities - Command | Vault | HashiCorp Developer

Question 2

Question Type: MultipleChoice

What are orphan tokens?

Options:

A- Orphan tokens are tokens with a use limit so you can set the number of uses when you create them

B- Orphan tokens are not children of their parent; therefore, orphan tokens do not expire when their parent does

C- Orphan tokens are tokens with no policies attached

D- Orphan tokens do not expire when their own max TTL is reached

Answer:

D

Explanation:

Orphan tokens are tokens that are root of their own token tree. This means that they do not have any parent token associated with them, and they do not expire when their parent token expires. Orphan tokens are useful for scenarios where you need a short-lived and independent token, such as for testing or debugging purposes. Orphan tokens can also be used to create temporary access tokens for applications or services that need to communicate with Vault without using a long-lived root token.Reference:Tokens | Vault | HashiCorp Developer,Vault cli: how to create orphan token with role - HashiCorp Discuss

Question 3

Question Type: MultipleChoice

Which of the following cannot define the maximum time-to-live (TTL) for a token?

Options:

A- By the authentication method t natively provide a method of expiring credentials

B- By the client system f credentials leaking

C- By the mount endpoint configuration very password used

D- A parent token TTL e password rotation tools and practices

E- System max TTL

Answer:

B

Explanation:

The maximum time-to-live (TTL) for a token is defined by the lowest value among the following factors:

The authentication method that issued the token. Each auth method can have a default and a maximum TTL for the tokens it generates. These values can be configured by the auth method's mount options or by the auth method's specific endpoints.

The mount endpoint configuration that the token is accessing. Each secrets engine can have a default and a maximum TTL for the leases it grants. These values can be configured by the secrets engine's mount options or by the secrets engine's specific endpoints.

A parent token TTL. If a token is created by another token, it inherits the remaining TTL of its parent token, unless the parent token has an infinite TTL (such as the root token). A child token cannot outlive its parent token.

System max TTL. This is a global limit for all tokens and leases in Vault. It can be configured by the system backend's max_lease_ttl option.

The client system that uses the token cannot define the maximum TTL for the token, as this is determined by Vault's configuration and policies. The client system can only request a specific TTL for the token, but this request is subject to the limits imposed by the factors above.

Question 4

Question Type: MultipleChoice

Which of these is not a benefit of dynamic secrets?

Options:

A- Supports systems which do not natively provide a method of expiring credentials

B- Minimizes damage of credentials leaking

C- Ensures that administrators can see every password used

D- Replaces cumbersome password rotation tools and practices

Answer:

C

Explanation:

Dynamic secrets are generated on-demand by Vault and have a limited time-to-live (TTL). They do not ensure that administrators can see every password used, as they are often encrypted and ephemeral. The benefits of dynamic secrets are:

They support systems that do not natively provide a method of expiring credentials, such as databases, cloud providers, SSH, etc. Vault can revoke the credentials when they are no longer needed or when the lease expires.

They minimize the damage of credentials leaking, as they are short-lived and can be easily rotated or revoked. If a credential is compromised, the attacker has a limited window of opportunity to use it before it becomes invalid.

They replace cumbersome password rotation tools and practices, as Vault can handle the generation and revocation of credentials automatically and securely. This reduces the operational overhead and complexity of managing secrets.

Question 5

Question Type: MultipleChoice

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?

Options:

A- Google Cloud Secrets Engine

B- Identity secrets engine

C- Key/Value secrets engine version 2

D- SSH secrets engine

Answer:

A

Explanation:

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.

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 6

Question Type: **MultipleChoice**

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"]
}
```

Options:

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

Answer:

C

Explanation:

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 7

Question Type: MultipleChoice

Which of the following statements describe the CLI command below?

S vault login -method-1dap username-mitche11h

Options:

A- Generates a token which is response wrapped

B- You will be prompted to enter the password

C- By default the generated token is valid for 24 hours

D- Fails because the password is not provided

Answer:

A

Explanation:

The CLI command vault login -method ldap username=mitchellh generates a token that is response wrapped. This means that the token contains a base64-encoded response wrapper, which is a JSON object that contains information about the token, such as its policies, metadata, and expiration time. The response wrapper is used to verify the authenticity and integrity of the token, and to prevent replay attacks. The response wrapper also allows Vault to automatically renew the token when it expires, or to revoke it if it is compromised. The -method ldap option specifies that the authentication method is LDAP, which requires a username and password to be provided. The username mitchellh is an example of an LDAP user name, and the password will be hidden when entered.Reference:Vault CLI Reference | Vault | HashiCorp Developer,Vault CLI Reference | Vault | HashiCorp Developer

Question 8

Question Type: MultipleChoice

What environment variable overrides the CLI's default Vault server address?

Options:

A- VAULT_ADDR

B- VAULT_HTTP_ADORESS

C- VAULT_ADDRESS

D- VAULT _HTTPS_ ADDRESS

Answer:

B

Explanation:

The environment variable VAULT_ADDR overrides the CLI's default Vault server address. The VAULT_ADDR environment variable specifies the address of the Vault server that is used to communicate with Vault from other applications or processes. By setting this variable, you can avoid hard-coding the Vault server address in your code or configuration files, and you can also use different addresses for different environments or scenarios. For example, you can use a local development server for testing purposes, and a production server for deploying your application.Reference:Commands (CLI) | Vault | HashiCorp Developer,Vault Agent - secrets as environment variables | Vault | HashiCorp Developer

Question 9

Question Type: MultipleChoice

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

Options:

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

Answer:

C

Explanation:

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.Reference:Dynamic secrets | Vault | HashiCorp Developer,What are dynamic secrets and why do I need them? - HashiCorp

Question 10

Question Type: MultipleChoice

What command creates a secret with the key "my-password" and the value "53cr3t" at path "my-secrets" within the KV secrets engine mounted at "secret"?

Options:

A- vault kv put secret/my-secrets/my-password 53cr3t

B- vault kv write secret/my-secrets/my-password 53cr3t

C- vault kv write 53cr3t my-secrets/my-password

D- vault kv put secret/my-secrets y-password-53cr3t

Answer:

A

Explanation:

The vault kv put command writes the data to the given path in the K/V secrets engine. The command requires the mount path of the K/V secrets engine, the secret path, and the key-value pair to store. The mount path can be specified with the -mount flag or as part of the secret path. The key-value pair can be given as an argument or read from a file or stdin. The correct syntax for the command is:

vault kv put -mount=secret my-secrets/my-password 53cr3t

or

vault kv put secret/my-secrets my-password=53cr3t

The other options are incorrect because they use the deprecated vault kv write command, or they have the wrong order or format of the arguments.Reference: https://developer.hashicorp.com/vault/docs/commands/kv/put3, https://developer.hashicorp.com/vault/docs/commands/kv4