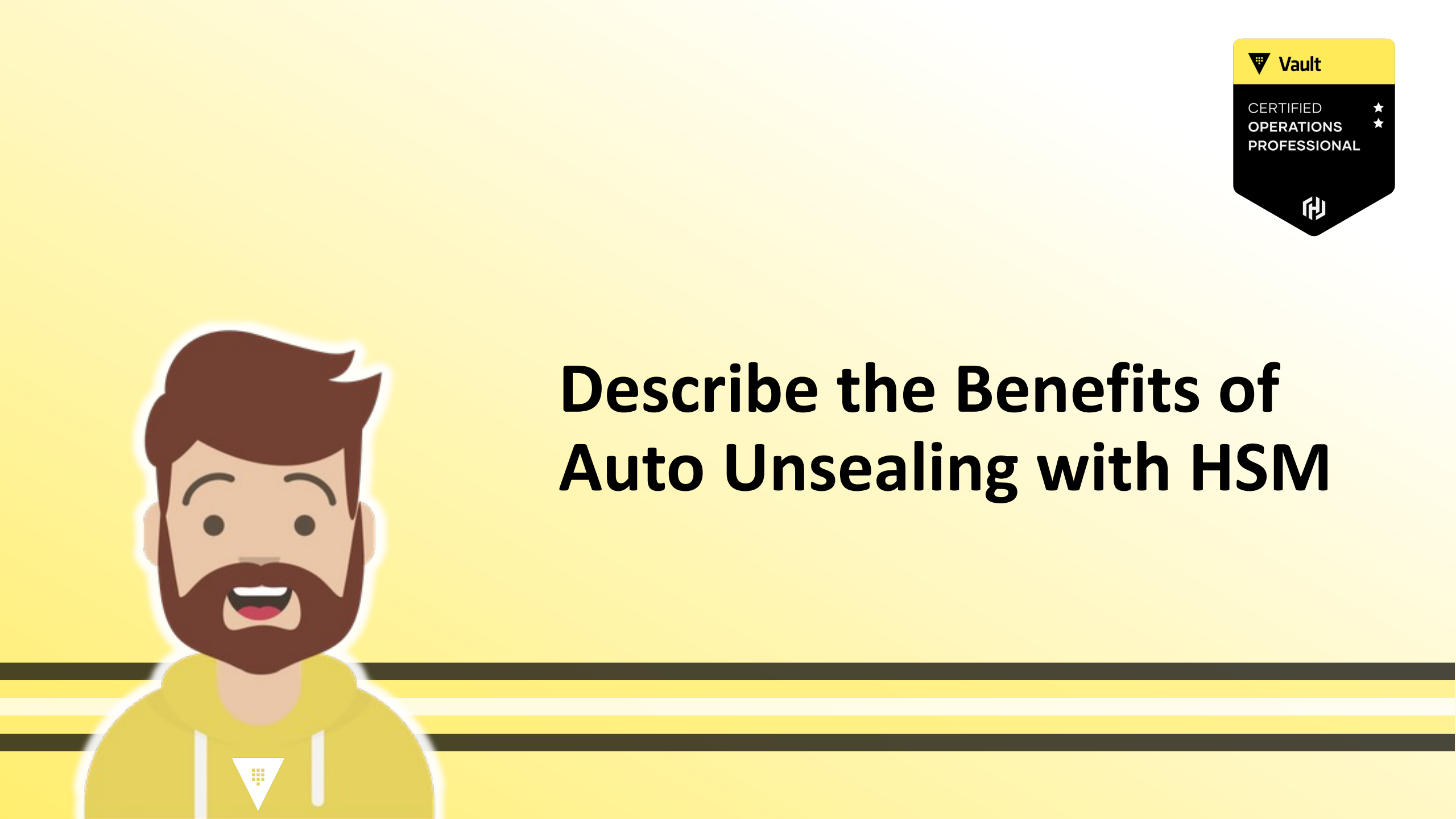


Describe the Benefits of Auto Unsealing with HSM

What is an HSM?

- An HSM is a network-based physical device that can safeguard and manage digital keys
- These keys can be used for encryption and decryption functions, digital signatures, strong authentication, or other functions
- HSMs commonly have tamper resistance – meaning that detection of tampering could invoke a response such as deleting the keys so nobody can access them

- Large enterprise customers often deploy dedicated physical HSMs in a traditional data center
- Public cloud providers offer access to dedicated or shared HSM services as well.
  - AWS CloudHSM or Azure Dedicated HSM is an HSM service where the HSM is dedicated to a single customer
  - AWS KMS is an example of a shared HSM service, where multiple customers may use a service that is backed by the same HSM

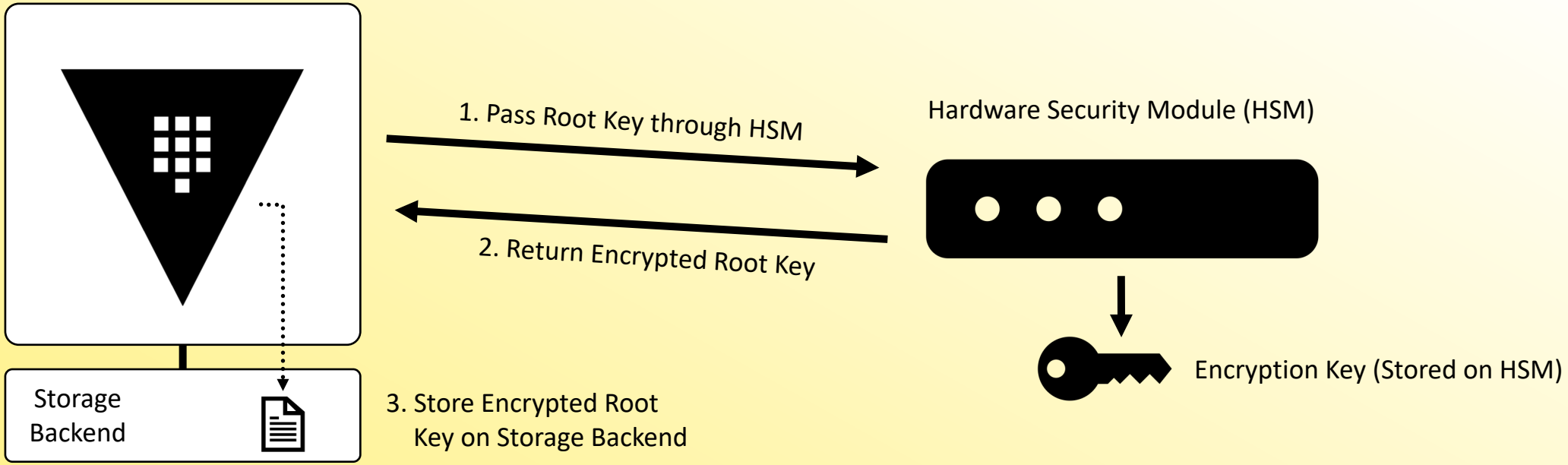
General HSM Support

Vault Enterprise has multiple integrations with an HSM:

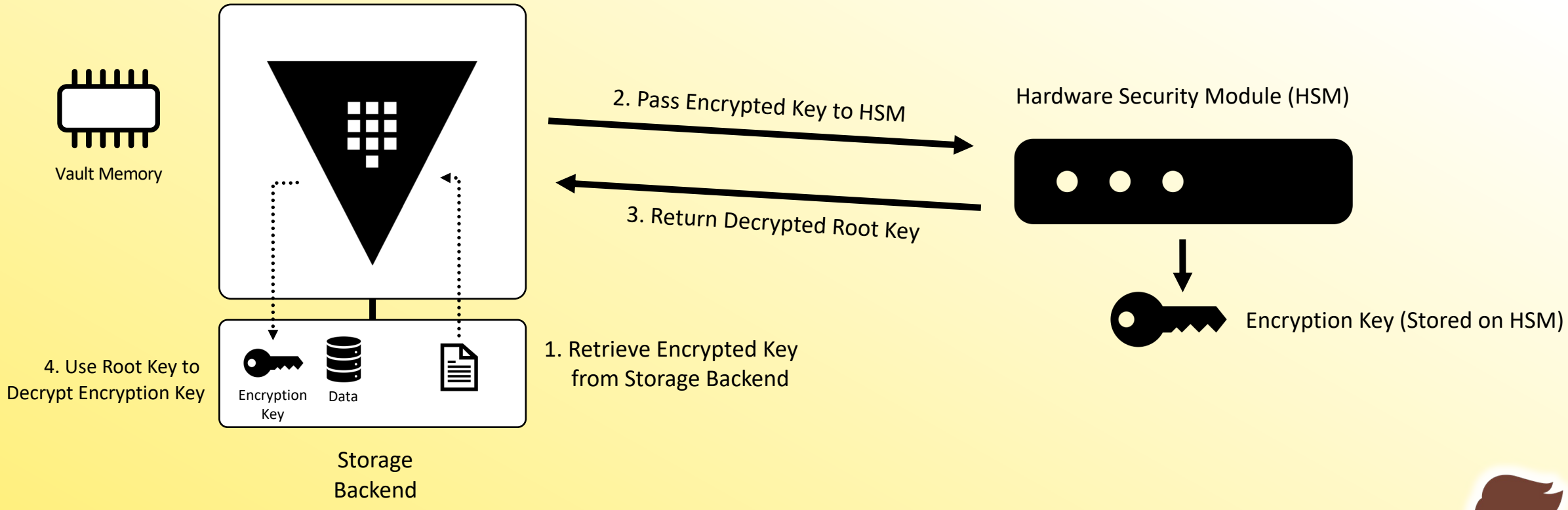
- Protect root key by using HSM to encrypt/decrypt root key
- Auto unseal Vault by storing wrapped key on local storage
- Seal wrapping to provide extra layer of protection for FIPS 140-2 compliance
- Entropy Augmentation to generate randomness for cryptographic operations

Requires HSM that supports PKCS11 standard

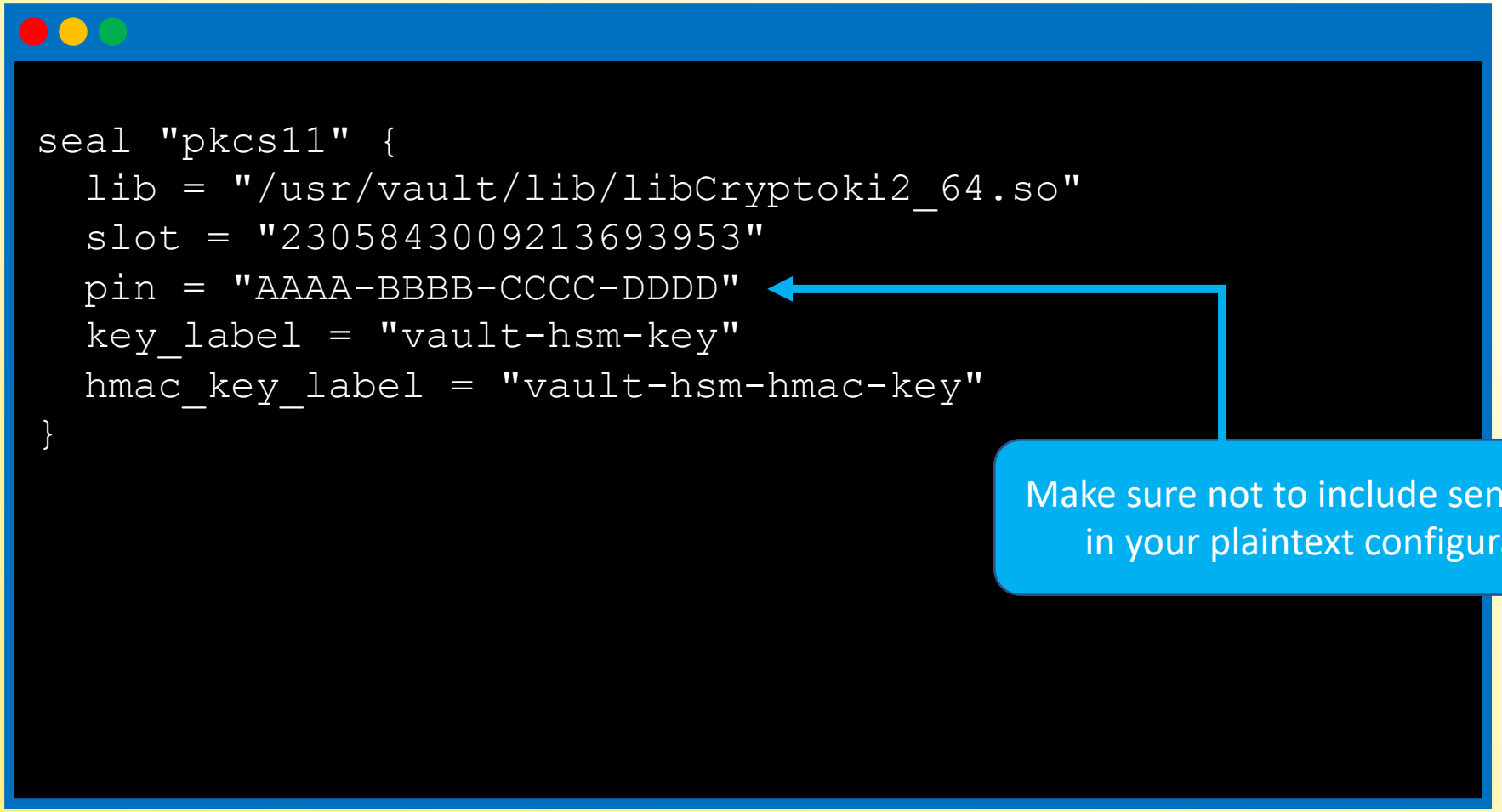
Initializing Vault



Auto Unseal with HSM



Configuration



Make sure not to include sensitive values in your plaintext configuration file

`pkcs11` Environment Variables

- VAULT_HSM_LIB
- VAULT_HSM_TYPE
- VAULT_HSM_SLOT
- VAULT_HSM_TOKEN_LABEL
- VAULT_HSM_PIN
- VAULT_HSM_KEY_LABEL
- VAULT_HSM_DEFAULT_KEY_LABEL
- VAULT_HSM_KEY_ID
- VAULT_HSM_HMAC_KEY_LABEL
- VAULT_HSM_HMAC_DEFAULT_KEY_LABEL
- VAULT_HSM_HMAC_KEY_ID
- VAULT_HSM_MECHANISM
- VAULT_HSM_HMAC_MECHANISM
- VAULT_HSM_GENERATE_KEY
- VAULT_HSM_RSA_ENCRYPT_LOCAL
- VAULT_HSM_RSA_OAEP_HASH
- VAULT_HSM_FORCE_RW_SESSION

You do NOT need to memorize these for the exam

Describe the Benefits and Use Cases of Seal Wrapping

What is Seal Wrapping?

Vault already protects my data using 256-bit AES, but how I can provide an extra layer of protection while meeting FIPS 140-2 compliance?

- Seal Wrapping essentially provides "double encryption" by encrypting the data using keys stored on an HSM
- Provides FIPS 140-2 compliance* by integrating with an HSM
  - Supports the FIPS level equivalent to the HSM – so if you use a Level 3 HSM, you will be used Level 3 cryptography
- Allows Vault to be deployed in high-security GRC environments (PCI, HIPAA, DoD, NATO)

*Starting with v1.10.3, HashiCorp is now publishing Vault binaries that can provide FIPS 140-2 compliance without requiring an HSM integration

What is Seal Wrapped by Default?

- Recovery Key
- Any stored key shares
- The root key
- The keyring

What Can We Enable?

- Seal wrapping is enabled by default on supported seals
- Causes values stored by the mount to be wrapped by the seal's encryption capability
  - You can disable this by setting `disable_sealwrap=true` in the config file

- Backend mounts (secrets engines, etc.) can take advantage of seal wrapping as well
- When enabling a secrets engine, provide the `seal_wrap=true` configuration
  - CLI flag to enable seal wrap on a secrets engine: `-seal-wrap`

Enabling Seal Wrapping for Key/Value


```
# Enable a secrets engine with seal wrap
$ vault secrets enable -seal-wrap kv


# List the enabled secrets engines
$vault secrets list -detailed
Path          Plugin     Accessor                Seal Wrap
----          ------     --------                -----------
...
cubbyhole/    cubbyhole  cubbyhole_b36dd7e1      false
identity/     identity   identity_b5650a96       false
kv/           kv         kv_fe02767b             true
```