Full credit is given to the above companies including the OS that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'clevis-encrypt-tpm2.1'

$ man clevis-encrypt-tpm2.1

CLEVIS-ENCRYPT-TPM(1) CLEVIS-ENCRYPT-TPM(1)

NAME

clevis-encrypt-tpm2 - Encrypts using a TPM2.0 chip binding policy

SYNOPSIS

clevis encrypt tpm2 CONFIG < PT > JWE

OVERVIEW

The clevis encrypt tpm2 command encrypts using a Trusted Platform Module 2.0 (TPM2) chip. Its only argument is the JSON configuration object.

When using the tpm2 pin, we create a new, cryptographically-strong, random key. This key is encrypted using the TPM2 chip. Then at decryption time, the key is decrypted again using the TPM2 chip.

```
$ clevis encrypt tpm2 '{}' < PT > JWE
```

The pin has reasonable defaults for its configuration, but a different hierarchy, hash, and key algorithms can be chosen if the defaults used are not suitable:

```
$ clevis encrypt tpm2 '{"hash":"sha1","key":"rsa"}' < PT > JWE
```

To decrypt the data, simply provide the ciphertext (JWE):

```
    $ clevis decrypt < JWE > PT
```

Note that like other pins no configuration is used for decryption, this is due clevis storing the public and private keys to unseal the TPM2 encrypted object in the JWE so clevis can fetch that information from there.

The pin also supports sealing data to a Platform Configuration Registers (PCR) state. That way the data can only be unsealed if the PCRs hashes values match the policy used when sealing.

For example, to seal the data to the PCR with index 0 and 1 for the SHA1 bank:

```
    $ clevis encrypt tpm2 '{"pcr_bank":"sha1","pcr_ids":"0,1"}' < PT > JWE
```

The PCR digest values are looked up from the current hash values for the PCRs, but a digest can also be provided if the data needs to be sealed with values different to the current ones, by passing the binary hash encoded in base64:

```
    $ clevis encrypt tpm2 '{"pcr_ids":"0","pcr_digest":"xy7J5svCtqlfM03d1lE5gdoA8MI"}' < PT > JWE
```

THREAT MODEL

The Clevis security model relies in the fact that an attacker will not be able to access both the encrypted data and the decryption key.

For most Clevis pins, the decryption key is not locally stored, so the decryption policy is only satisfied if the decryption key can be remotely accessed. It could for example be stored in a remote server or in a hardware authentication device that has to be plugged into the machine.

The tpm2 pin is different in this regard, since a key is wrapped by a TPM2 chip that is always present in the machine. This does not mean that there are not use cases for this pin, but it is important to understand the fact that an attacker that has access to both the encrypted data and the local TPM2 chip will be able to decrypt the data.

CONFIG

This command uses the following configuration properties:

? hash (string) : Hash algorithm used in the computation of the object name (default: sha256)

It must be one of the following:

- ? sha1
- ? sha256
- ? sha384
- ? sha512
- ? sm3_256

? key (string) : Algorithm type for the generated key (default: ecc)

It must be one of the following:

- ? rsa
- ? keyedhash
- ? ecc
- ? symcipher

? pcr_bank (string) : PCR algorithm bank to use for policy (default: sha1)

It must be one of the following:

- ? sha1
- ? sha256

? pcr_ids (string) : Comma separated list of PCR used for policy. If not present, no policy is used

? pcr_digest (string) : Binary PCR hashes encoded in base64. If not present, the hash values are looked up

SEE ALSO

clevis-decrypt(1)