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

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'clevis-luks-bind.1' command

$ man clevis-luks-bind.1

CLEVIS-LUKS-BIND(1) CLEVIS-LUKS-BIND(1)

NAME

clevis-luks-bind - Bind a LUKS device using the specified policy

SYNOPSIS

clevis luks bind [-f] [-y] -d DEV [-t TKN_ID] [-s SLT] [-k KEY] [-e EXISTING_TOKEN_ID] PIN CFG

OVERVIEW

The clevis luks bind command binds a LUKS device using the specified policy. This is accomplished with a simple command:

$ clevis luks bind -d /dev/sda tang '{"url":...}'

This command performs four steps:

1. Creates a new key with the same entropy as the LUKS master key ? maximum entropy bits is 256.
2. Encrypts the new key with Clevis.
3. Stores the Clevis JWE in the LUKS header.
4. Enables the new key for use with LUKS.

This disk can now be unlocked with your existing password as well as with the Clevis policy. You will additionally need to enable one or more of the Clevis LUKS unlockers. See clevis-luks-unlockers(7).

OPTIONS

? -f : Do not prompt for LUKSMeta initialization

? -y : Automatically answer yes for all questions. When using tang, it causes the advertisement trust check to be skipped, which can be

useful in automated deployments

? -d DEV : The LUKS device on which to perform binding

? -s SLT : The LUKSMeta slot to use for metadata storage

? -t TKN_ID : The LUKS token ID to use; only available for LUKS2

? -k KEY : Non-interactively read LUKS password from KEY file

? -k - : Non-interactively read LUKS password from standard input

? -e E_TKN_ID : LUKS token ID for existing passphrase; only available for LUKS2. This parameter allows providing a configured token ID in LUKS2 containing the existing passphrase for this device, so that existing passphrase is not prompted by clevis

CAVEATS

This command does not change the LUKS master key. This implies that if you create a LUKS-encrypted image for use in a Virtual Machine or Cloud environment, all the instances that run this image will share a master key. This is extremely dangerous and should be avoided at all cost. This is not a limitation of Clevis but a design principle of LUKS. If you wish to have encrypted root volumes in the cloud, you will need to make sure that you perform the OS install method for each instance in the cloud as well. The images cannot be shared without also sharing a master key.

SEE ALSO

clevis-luks-unlockers(7), clevis-encrypt-tang(1), clevis-encrypt-sss(1), clevis-decrypt(1)