cryptsetup

Gérer les volumes chiffré dm-crypt et LUKS

cryptsetup est utilisé pour définir les mappages device-mapper gérés par dm-crypt. Cela inclus les volume dm-crypt et LUKS. La différence est que LUKS utilise en en-tête de méta-données et peut ainsi offrir plus de fonctionnalités que dm-crypt. D'un autre côté, l'en-tête est visible et vulnérable.

De plus, cryptsetup fournis un support limité des volumes historiques loopaes et pour les volumes compatibles TrueCrypt.

Beaucoup d'informations sur les riques de l'utilisation de stockage chiffré, la gestion des problèmes et sur les aspects de sécurité peuvent être trouvés dans la FAQ Cryptsetup.

Commandes de base

open <device> <name> –type <device_type> Ouvre (créé un mappage avec) le nom. device_type peut être plain, luks, loopaes ou tcrypt.

close <name> Supprime la mappage existant et détruit la clé de la mémoire kernel

status <name> Reporte le status pour la mappage donné

resize <name> Redimentionne un mappage active

Mode Plain

plain dm-crypt chiffre le périphérique secteur par secteur avec un simple, non-salted hash de la passphrase. aucune vérification n'est effectuée, aucune métadonnée n'est utilisée. Il n'y a pas d'opération de formattage. Quand le périphérique brut est mappé, les opérations de périphériques peuvent être utilisée sur le périphérique mappé, incluant la création du système de fichier. Les périphériques de mappage peuvent être créés dans /dev/mapper/<name>.

Extensions LUKS

LUKS est un standard pour le chiffrement de disque. Il ajoute une en-tête standardisé au début du périphérique, une zone de slot directement derrière l'en-tête et les données bulk ensuite. Toute ce jeu est appelé un conteneur LUKS. Le périphérique où un conteneur LUKS réside est appelé un périphérique LUKS.

LUKS peut gérér plusieurs passphrases qui peuvent être révoqués individuellement ou changés et peuvent être néttoyés du média persistant de manière sécurisée. Les passphases sont protégées contre le brute-force et les attaques par dictionnaire par PBKDF2, qui implémente une itération de hash et un salt dans une fonction.

Chaque passphrase, également appelée une clé, est associée avec un des 8 slots. Les opérations de clé qui ne spécifient pas un slot affectent le premier slot qui matche la passphrase fournie ou le premier slot vide si une nouvelle passphrase est ajoutée.

Le paramètre device peut également être spécifié par un UUID LUKS au format UUID=<uuid>. Pour spécifier un en-tête détaché, le paramètre –header peut être utilisé dans toutes les commandes LUKS et prend toujours précédence sur le paramètre positionnel device.

Les options LUKS valides sont les suivantes :

luksFormat <device> [<key file>] Initialise une partition LUKS et définis la passphrase initiale (slot 0), soit en demandant, soit via le fichier spécifié. Noter que si le 2ème argument est présent, la passphrase est prise du fichier donné, sans avoir besoin de l'option –key-file. Noter également que '-' comme nom de fichier lit la passphrase depuis l'entrée standard. cette action ne peut être utilisé que sur des périphériques LUKS qui ne sont pas mappé.

open –type luks <device> <name> Ouvre le périphérique LUKS et définis un mappage une fois la vérification de la passphrase effectuée.

luksSuspend <name> Suspend un périphérique actif (toutes les opérations IO seront blockés et les accès au périphérique attendent indéfiniment.

luksResume <name> Résume un périphérique suspendu et redemande une passphrase, si –key-file n'est pas donné

luksAddKey <device> [<key file with new key>] Ajoute une nouvelle passphrase. Une passphrase doit être fournie interactivemenet ou via –key-file.

luksRemoveKey <device> [<key file with passphrase to be removed>] Supprime la passphrase fournie du périphérique LUKS. La passphrase à supprimer peut être spécifiée interactivement ou via –key-file

luksChangeKey <device> [<new key file>] Change une passphrase existante.

luksKillSlot <device> <key slot number> Détruit la clé du périphérique LUKS.

luksErase <device> Supprime tous les keyslots et rend le conteneur inaccessible. Cette opération est irréversible

luksUUID <device> Affiche le UUID du périphérique LUKS. Définis un nouvel UUID is –uuid est spécifié

isLuks <device> Retourne true, si le périphérique est un périphérique LUKS.

luksDump <device> Dump les informations d'en-tête d'une périphériques LUKS. Si l'options –dump-master-key est utilisé, la clé maître est dumpé au lieu du keyslot.

luksHeaderBackup <device> –header-backup-file <file> Stocke un backup binaire de l'en-tête LUKS et la zone keyslslot.

luksHeaderRestore <device> –header-backup-file <file> Restaure un backup binaire d'un en-tête LUKS.

Extensions TCRYPT

cryptsetup supporte le mappage de TrueCrypt, tcplay ou VeraCrypt en utilisant l'API kernel Linux. Le changement de formattage d'en-tête et l'en-tête TCRYPT n'est pas supporté.

L'extension TCRYPT nécessite l'API crypto disponible dans l'espace utilisateur. (CRYPTO_USER_API_SKCIPHER) Parce que l'en-tête TCRYPT est chiffré, il faut toujours fournir une passphrase valide.

cryptsetup devrait reconnaître toutes les variantes d'en-tête, excepté les chaînes de chiffrement utilisant le mode de chiffrement LRW avec block de chiffrement 64-bits (blowfish en mode LRW n'est pas reconnu, c'est une limitation de l'API crypto du kernel)

Parce que l'en-tête TCRYPT est chiffré, il faut toujours fournir une passphrase valide et un keyfile

Pour reconnaître un périphérique VeraCrypt, utiliser l'option –veracrypt. VeraCrypt est une extension de l'en-tête TrueCrypt avec un compteur d'itération amélioré, donc le déblocage peut prendre plus de temps.

Note : L'activation avec tcryptOpen est supporté uniquement pour les chaînes de chiffrement utilisant les modes LRW ou XTS.

tcryptDump devrait fonctionner pour tous les périphériques TCRYPT reconnus et ne nécessite pas de privilège root.

Pour mapper les périphériques système (avec un boot loader) utiliser l'option –tcrypt-system.

Si vous avez un périphérique TCRYPT comme fichier image et souhaitez mapper plusieurs partitions chiffrées avec le chiffrement système, créer un mappage loopback avec les partitions en premier (losetup -P), et utiliser la partition loop comme paramètre de périphérique.

Pour utiliser un en-tête caché, utiliser –tcrypt-hidden. Pour utiliser en-tête backup, utiliser l'option –tcypt-hidden.

open –type tcrypt <device> <name> Ouvre un périphérique TCRYPT et définis le mappage. keyfile permet de combiner le contenu du fichier avec la passphrase et peut être répétée. Noter qu'utiliser des keyfiles est compatible avec TCRYPT et est différent de la logique keyfile LUKS.

tcryptDump <device> Dump les informations d'en-tête. Si –dump-master-key est utilisé, la clé maître est dumpé au lieu de l'en-tête.

Divers

repair <device> Tente de réparer les méta-données du périphérique. Uniquement pour les périphériques LUKS

benchmark <options> benchmark les chiffrements et les fonction de dérivations de clé (KDF). Sans paramètres, tente de mesurer les configuration communes. Les paramètre –cipher, –key-size ou –hash doivent être spécifiés.

OPTIONS

-v, –verbose mode verbeux

–debug Mode debug

-h, –hash Spécifie le hash de la passphrase à ouvir

-c, –cipher définis le chiffrement

-y, –verify-passphrase Demande 2 fois la passphrase

-d, –keyfile lit la passphrase depuis le fichier

–keyfile-offset Saut n octets au début du fichier de clé

-l, –keyfile-size Taille de la clé en octets dans le fichier de clé

–new-keyfile-offset Saute n octets en ajoutant une nouvelle passphrase dans le fichier de clé avec luksAddKey.

–new-keyfile-size Taille de la clé en octet en ajoutant une nouvelle passphrase dans le fichier de clé avec luksAddKey.

–master-key-file Utilise la clé maître stockée dans un fichier. Pour luksFormat, cela permet de créer un en-tête LUKS avec cette clé maître.

–dump-master-key Pour luksDump, inclus la clé maître dans les informations affichées.

–use-random, –use-urandom Spécifie le générateur de nombre pseudo-aléatoire utilisé pour clé la clé de volume

-S, –key-slot Spécifie le slot de clé à utiliser. Tous les autres slots seront désactivé.

-b, –size Force la taile du périphérique en secteurs de 512 octets.

-o, –offset Décalage du début dans le périphérique en secteurs de 512 octets

-p, –skip Saut n secteurs de 512 octets dans le périphérique.

-r, –readonly Définis un mappage lecture seule

–shared Créé un mappage additionnel pour un périphérique ciphertext commun.

–i, –iter-time Nombre de temps en ms pour le traitement PBKDF2.

-q, –batch-mode mode silencieux

-t, –timeout Temps d'attente de la passphrase

-T, –tries Nombre de tentative pour l'entrée de passphrase invalide

–align-payload Aligne le payload au limites de n secteurs de 512 octets.

–uuid Utilise l'UUID fournis au lieu d'en générer un nouveau

–allow-discards Autorise l'utilisation des requêtes discards (TRIM) déconseillé pour des raisons de sécurité

–perf-same_cpu_crypt Effectue un chiffrement avec le même CPU qui gère les E/S.

–perf-submit_from_crypt_cpus Désactive les écriture offload dans un thread séparé après le chiffrement.

–test-passphrase N'active pas le périphériques, vérifie simplement la passphrase

–header Utilise un périphérique de méta-donnée séparé ou un fichier où se trouve l'en-tête LUKS

–force-password N'utilise pas la vérification du mot de passe LUKS

Codes de retour

0 L'opération s'est déroulé avec succès

1 Mauvais paramètres

2 N'a pas les permissions

3 Out of memory

4 Mauvais périphérique spécifié

5 Le périphérique existe déjà