dm-crypt

Chiffrement transparent des périphériques block

la cible crypt de Device-Mapper fournis un chiffrement transparent des périphériques blocks utilisant l'API crypto du kernel

Paramètres

cipher Algorithme de chiffrement et un mode de génération IV optionel (ex : des, aes-cbc-essip :sha256, twofish-ecb). /proc/crypto contient les modes crypto supportés

key Clé utilisée pour le chiffrement. Encodé en hexadécimal.

keycount Mode compatibilité multi-clé. On peut définis keycount clé et les secteurs sont chiffrés en accord avec leur offset (le secteur 0 utilise key0, etc.)

iv_offset L'offset IV est un compteur de secteur qui sont ajoutés au numéro du secteur avant de créer l'IV

device_path Périphérique qui est utilisé comme backend et contient la donnée chiffrée.

offset Secteur de départ dans le périphérique où les données chiffrée commencent

#opt_params Nombre de paramètres optionnels.

Paramètres optionnels

allow_discards Block discard request passé au périphérique crypté. Défaut : ignore ces requêtes

same_cpu_crypt Effectue le chiffrement en utilisant le même cpu que l'IO qui l'a émis.

submit_from_crypt_cpus Désactive l'écriture offload dans un thread séparé après le chiffrement. L'écriture bénéficie à CFQ, mais peut dégrader les performances dans certains cas.

Scripts d'exemple

LUKS (Linux Unified Key Setup) est la manière préférée pour définir le chiffrement de disque avec dm-crypt en utilisant l'utilitaire cryptsetup

```
[[
#!/bin/sh
# Crée un périphérique crypto avec dmsetup
dmsetup create crypt1 -table "0 `blockdev -getsize $1` crypt aes-cbc-essiv:sha256
babebabebabebabebabebabebabebabe 0 $1 0"
]]

[[
#!/bin/sh
# Crée un périphérique en utilisant cryptsetup et LUKS avec le chiffrement par défaut:
cryptsetup luksFormat $1
cryptsetup luksOpen $1 crypt1
]]
```