keys-ecryptfs

Système de fichier stacké à chiffrement transparent

ecryptfs est un système de fichier stacké qui chiffre et déchiffre de manière transparente chaque fichier en utilisant une clé de chiffrement de fichier (FEK) générée aléatoirement.

Chaque FEK est en retour chiffré avec une clé de chiffrement de clé de chiffrement de fichier (FEFEK) soit dans l'espace kernel ou en espace utilisateur avec un service appelé ecryptfsd. Dans le premier cas, l'opération est effectuée directement par le cryptoAPI du kernel en utilisant une clé, la FEFEK, dérivée d'une passphrase de l'utilisateur ; dans l'autre cas, la FEK est chifrée par ecryptfsd avec l'aide de librairies externes pour supporter d'autres mécanismes comme les clés publiques, PKCS#11 et TPM.

La structure de données définie par ecryptfs pour contenir les informations requises pour le déchiffrement FEK est appelé le jeton d'authentification et , actuellement, peut être stocké dans une clé kernel de type 'user', inséré dans la session de l'utilisateur par l'utilitaire userspace mount.ecryptfs fournis avec le paquet ecryptfs-utils

Le type de clé encrypted a été étendue avec l'introduction du nouveau format ecryptfs pour être utilisé en conjonction avec le système de fichier ecryptfs. Les clé chiffrées au nouveau format stockent un jeton d'authentification dans son payload avec un FEFEK générée aléatoirement par le kernel et protégé par la clé maître.

Pour éviter les attaques plaintext, le 'datablob' obtenu via les commandes 'keyctl print' ou 'keyctl pipe' ne contiennent pas tout le jeton d'authentification, dont le contenu est connu, mais seulement la FEFEK sous forme chiffrée.

Le système de fichier ecryptfs peut réellement bénéficier de l'utilisation de clé chiffrées car les clés requises peuvent être générées de manière sécurisée par un administrateur et fournis au boot une fois la clé de confiance fournie pour effectuer le montage dans un environnement contrôlé. Un autre avantage est que la clé n'est pas exposée aux traitements de logiciels malicieux, parce qu'elle est disponible en clair seulement au niveau du kernel.

Utilisation

keyctl add encrypted name "new ecryptfs key-type :master-key-name keylen" ring
keyctl add encrypted name "load hex_blob" ring
keyctl update keyid "update key-type :master-key-name"
name := '< 16 hexadecimal characters >'
key-type := 'trusted' | 'user'
keylen := 64

Exemple d'utilisation de clé chiffrée avec le système de fichier ecryptfs

Créer un clé chiffrée "1000100010001000" de 64bits de long au format ecryptfs et la sauver avec une clé utilisateur précédemment chargée "test" :
keyctl add encrypted 1000100010001000 "new ecryptfs user :test 64" @u 19184530
keyctl print 19184530
ecryptfs user :test 64 490045d4bfe48c99f0d465fbbbb79e7500da954178e2de0697
dd85091f5450a0511219e9f7cd70dcd498038181466f78ac8d4c19504fcc72402bfc41c2
f253a41b7507ccaa4b2b03fff19a69d1cc0b16e71746473f023a95488b6edfd86f7fdd40
9d292e4bacded1258880122dd553a661

keyctl pipe 19184530 > ecryptfs.blob
Monter un système de fichier ecryptfs avec la clé chiffrée créée "1000100010001000" dans le répertoire /secret :
mount -i -t ecryptfs -oecryptfs_sig=1000100010001000,ecryptfs_cipher=aes,ecryptfs_key_bytes=32 /secret /secret