DM-Crypt

Położenie: (nie dotyczy)

Ogólnie

Nazwa *DM-Crypt* pochodzi od „*device mapper crypt*”. Funkcja ta zaimplementowana jest w sa mym jądrze Linuksa. Mechanizm oparty jest na szyfrowaniu symetrycznym (czyli zaszyfrowane dane chronione są hasłem; możliwa jest także ochrona za pomocą klucza w pliku). Aby wszystko działało, potrzebujemy następujących modułów:

Device Drivers →
[*] Block devices
    <> Loopback device support* (moduł *loop*; u mnie jest na stałe w jądrze)
[*] Multiple devices driver support (RAID and LVM):
    <M> Device mapper support (moduł *dm_mod*)
        <M> Crypt target support (moduł *dm_crypt*)
Cryptographic API →
    wybieramy metody szyfrowania, które chcemy używać (jako moduły)

Moduł *dm_mod* ładowany jest automatycznie przez *udev*, zaś moduł *dm_crypt* jest ładowany tylko w momencie jego używania.

Do obsługi modułu *dm_crypt* potrzebna będzie instalacja odpowiedniego pakietu:
emerge -vp cryptsetup (wszystkie operacje można wykonać za pomocą *dmsetup*, ale jest to dosyć skomplikowane; plik konfiguracyjny: */etc/conf.d/dmcrypt*)
rc-update add dmcrypt boot

Tworzenie szyfrowanej partycji

cryptsetup --verify-passphrase create mapowaneUrządzenie /dev/sdb1 (tworzymy ma powane urządzenie w */dev/mapper/* z dwukrotnym podaniem hasła; w chwili obecnej, nie ma możliwości późniejszej zmiany hasła; aby wymusić konkretny algorytm szyfrowania, używamy pa rametru „*-c nazwaAlgorytmu*”; dostępne nazwy algorytmów otrzymamy po wydaniu komendy „*cat /proc/crypto*”; domyślnym algorytmem jest AES)
dmsetup ls (sprawdzamy czy mapowane urządzenie zostało na pewno utworzone)
mkreiserfs /dev/mapper/mapowaneUrządzenie (formatujemy partycję na zmapowanym urządzeniu)
cryptsetup remove mapowaneUrządzenie (likwidujemy zmapowane urządzenie)

Praca z szyfrowaną partycją

cryptsetup create mapowaneUrządzenie /dev/sdb1 (ponownie tworzymy mapowane urządzenie; jego nazwa może być inna, ale musimy podać prawidłowe hasło, inaczej zostanie utworzone urządzenie, ale niemożliwe będzie jego zamontowanie)
 # mount /dev/mapper/mapowaneUrządzenie /mnt/cypher (montujemy zmapowane urzą dzenie; tylko root ma prawa zapisu na zamontowanym urządzeniu)
Po wykonaniu pracy na szyfrowanej partycji:
umount /mnt/cypher
cryptsetup remove mapowaneUrządzenie

Tworzenie szyfrowanego kontenera

touch /mnt/usb/kontener.txt (tworzymy pusty plik kontenera)
shred -n1 -s500M /mnt/usb/kontener.txt (nadajemy mu odpowiednią wielkość i jednorazo wo wypełniamy losowymi danymi; program *shred* służy do nadpisywania plików w celu zabezpie czenia ich przed wydobyciem danych)
losetup /dev/loop0 /mnt/usb/kontener.txt (wiążemy urządzenie *loopback* z utworzonym kontenerem)
cryptsetup --verify-passphrase create mapowaneUrządzenie /dev/loop0 (tworzymy mapowane urządzenie kontenera oraz hasło dostępu; aby wymusić konkretny algorytm szyfrowa nia, używamy parametru „*-c nazwaAlgorytmu*”; dostępne nazwy algorytmów otrzymamy po wy

daniu komendy „*cat /proc/crypto*”; domyślnym algorytmem jest AES)
dmsetup ls (sprawdzamy, czy wszystko jest OK)
mkreiserfs /dev/mapper/mapowaneUrządzenie (formatujemy kontener)
cryptsetup remove mapowaneUrządzenie
losetup -d /dev/loop0 (likwidujemy powiązanie *loopback* z kontenerem)

Praca z szyfrowanym kontenerem

losetup /dev/loop0 /mnt/usb/kontener.txt (wiążemy urządzenie *loopback* z utworzonym kontenerem)
cryptsetup create mapowaneUrządzenie /dev/loop0 (tworzymy mapowane urządzenie kontenera i odbiezpieczamy je naszym hasłem)
mount /dev/mapper/mapowaneUrządzenie /mnt/cypher (montujemy zmapowane urzą dzenie; tylko root ma prawa zapisu na zamontowanym urządzeniu)
Po zakończonej pracy:
umount /mnt/cypher
cryptsetup remove mapowaneUrządzenie
losetup -d /dev/loop0 (likwidujemy powiązanie *loopback* z kontenerem)

Ważne: program *cryptsetup* posiada rozszerzenie LUKS (*Linux Unified Key Setup*), które umożliwia tworzenie więcej niż jednego hasła (np. dla innych użytkowników) i zarządzanie tymi hasłami. Więcej po wydaniu komendy:
cryptsetup --help
cryptsetup luksDump /dev/mapper/mapowaneUrządzenie (wykaz haseł)

Ostatnia aktualizacja: 4 kwietnia 2014.