Cifrare FileSystem in Linux

Nell'articolo precedente abbiamo visto come cifrare lo swap usando `cryptsetup` in Linux. Vogliamo ora invece cifrare una partizione dati. Per fare questo utilizziamo i comandi LUKS (Linux Unified Key Setup) di `cryptsetup` per gestire la partizione virtuale cifrata.

Cryptsetup-luks

Cryptsetup-luks permette di gestire in maniera avanzata le passphrase utilizzate per accedere ai filesystem cifrati. Il processo utilizzato da cryptsetup-luks è il seguente. Il filesystem è cifrato con una master-key generata automaticamente in modo pseudo-casuale al momento della creazione iniziale del filesystem cifrato. La master-key viene poi scritta in forma cifrata, con lo stesso algoritmo con cui è cifrato il filesystem, negli header della partizione. Esistono 8 slot ciascuno dei quali può contenere una versione della master-key cifrata con una passphrase diversa, almeno uno degli 8 slot deve essere occupato. E' possibile quindi utilizzare sino a 8 diverse passphrase per accedere al filesystem, e questo sistema permette anche di cambiare passphrase semplicemente creandone una nuova e cancellando quella vecchia. La master-key non è però cifrata direttamente con la passphrase, ma con una chiave generata dalla passphrase con il processo di hashing detto PBKDF2 [2], questo per prevenire possibili attacchi di dizionario alle master-key cifrate dovuti alla debolezza intrinseca delle passphrase. Ulteriori accorgimenti sono stati presi per diminuire le possibilità di vari attacchi alle master-key cifrate quali procedure anti-forensi [1].

La procedura di utilizzo

Il filesystem cifrato deve essere creato dall'amministratore di sistema (*root*), mentre per le usuali operazioni di utilizzo l'amministratore può creare degli script e utilizzare strumenti quali `sudo` per permettere agli utenti di montare/smontare autonomamente i filesystem cifrati. Queste procedure non sono indicate nei nostri esempi, ma supponiamo che l'utente abbia sufficienti privilegi per poter

eseguire i comandi.

In Tabella 1 sono riportati i principali comandi per la creazione di un filesystem cifrato sulla partizione `/dev/hdd1`, che ovviamente deve essere vuota di ogni contenuto. Prima di tutto, con il comando `dd` la partizione viene riempita di dati casuali (questo comando può richiedere parecchio tempo per essere eseguito). La casualità dei dati iniziali è molto importante per proteggersi da possibili attacchi crittografici.

Poi il primo comando `cryptsetup` crea la tavola della partizione cifrata nel device `/dev/hdd1`, con l'algoritmo AES a 256bit[1], e chiede la passphrase con cui viene cifrata la master-key e scritta negli header della partizione nello slot numero 0. Il secondo comando `cryptsetup` monta la partizione cifrata sul device `/dev/mapper/dm-crypt-dev`, questo può essere controllato con il comando `cryptsetup status dm-crypt-dev`. Come abbiamo indicato nell'articolo precedente, scopo di questo device virtuale è quello di de/cifrare tutti i dati che lo attraversano in modo che sul disco i dati siano sempre cifrati, mentre l'utente possa accedervi in chiaro. Si noti come quest'ultimo comando `cryptsetup` riporta il numero di slot in cui è posizionata la master-key decifrata, è importante notare il numero di slot poiché sarà poi necessario quando si vorrà cambiare o cancellare questa passphrase.

I successivi comandi creano un filesystem nella partizione cifrata (per semplicità lo abbiamo scelto di tipo ext3, default in Linux) e poi montano il nuovo filesystem sulla directory `/home/user/dm-crypt-dir`, dando anche la proprietà della root directory interna al filesystem all'utente. A questo punto l'utente può accedere ai propri dati contenuti nella partizione cifrata `/dev/hdd1` tramite la directory `/home/user/dm-crypt-dir`.

Nelle Tabelle 2 e 3 sono riportati i comandi che l'utente successivamente deve utilizzare per montare e smontare il filesystem cifrato. Come abbiamo detto, questi comandi richiederebbero il privilegio di amministratore per essere eseguiti, ma l'amministratore può permettere, con usuali configurazioni dei sistemi operativi UNIX, all'utente di eseguirli direttamente. Si noti che solo il comando

1 Per maggiore protezione contro gli attacchi di watermarking e simili, si consiglia di usare l'algoritmo aes-cbc-essiv:sha256 che adotta una gestione più sofisticata degli IV.

`cryptsetup luksOpen` richiede la passphrase necessaria per decifrare la master-key. In Tabella 4 sono riportati i comandi per creare una nuova passphrase o cancellarne una vecchia, ove `NUM` è il numero di slot ove è la passphrase che si vuole cancellare. Ovviamente la creazione di una nuova passphrase richiede la verifica di una passphrase precedente anche perché deve essere decifrata la master-key per poter essere cifrata con la nuova passphrase. Si noti che se si cancellano tutte le passphrase, tutti i dati sono persi in quanto non è più possibile recuperare la master-key (ovviamente il sistema avverte prima di eseguire questo il comando).

Usare un file come contenitore di un filesystem cifrato

Invece di usare una partizione fisica del disco, è anche possibile creare un file che contenga la partizione cifrata. Per fare questo utilizziamo oltre a `cryptsetup` anche `losetup` che permette di utilizzare un file come se fosse una partizione.

In Tabella 5 sono riportati i principali comandi per la creazione di un filesystem cifrato di 100MB, come indicato dal parametro `DimMB`. Con il comando `dd` viene quindi creato un file di 100MB, `/home/user/dm-crypt-file`, riempito di dati casuali. Con il comando `losetup` questo file viene montato come se fosse un device sul device virtuale `/dev/loop0`. Poi si possono usare i comandi indicati nelle Tabelle 1, 2 e 3 (eccetto il comando `dd` ovviamente) sostituendo `/dev/hdd1` con `/dev/loop0`. In ogni caso prima dei comandi `cryptsetup` per montare la partizione bisogna utilizzare il comando `losetup` per connettere `/dev/loop0` al file, e viceversa dopo aver smontato la partizione cifrata bisogna usare `losetup` per rilasciare il device `/dev/loop0`.

E' necessaria una avvertenza: quando si usa un filesystem cifrato basato su di un file, è meglio che il filesystem di base in cui è il file contenitore, non sia di tipo *journalling,* altrimenti potrebbe essere possibile che diverse copie di settori del file contenitore cifrato siano presenti sul disco, dando la possibilità ad un crittoanalista di mettere in atto alcuni tipi di attacchi.

Alcune considerazioni

Il sistema appena descritto permette di gestire le passphrase e cifrare filesystem, sia interi dischi che partizioni di dischi che partizioni basate su file. Nel caso di partizioni basate su file vi è anche la possibilità di spostare/copiare il filesystem cifrato su altri elaboratori, anche con sistemi operativi diversi. Ad esempio FreeOTFE (Free On-The-Fly Encryption) per sistemi operativi Microsoft è compatibile con cryptsetup-luks.

D'altra parte una partizione od un file contenente una partizione cifrata con cryptsetup-luks è facilmente riconoscibile data la particolare struttura degli header della partizione fisica su disco che deve contenere la master-key cifrata con le varie passphrase. L'utilizzo di cryptsetup senza luks, che avevamo utilizzato per cifrare lo swap nel precedente articolo (le due modalità operative si differenziano solo per le opzioni passate al comando), non memorizza su disco la chiave di cifratura ed a prima vista sembrerebbe più sicura sia perché di fatto meno riconoscibile sia perché la chiave di cifratura non risiede nella stessa partizione con i dati. In realtà è ben noto che la gestione manuale delle passphrase o delle chiavi di cifratura è soggetta a molti rischi, e con cryptsetup senza luks non è possibile cambiare la chiave (l'unica possibilità è creare un'altra partizione cifrata e copiarvi i dati della precedente), mentre cryptsetup-luks permette di cambiare la passphrase quando si vuole. Cryptsetup-luks permette inoltre di generare passphrase tenute di riserva o per key-escrow, cosa altrimenti non possibile. In conclusione il livello di sicurezza nell'utilizzo dei due modi di cryptsetup dipende molto dagli scopi e dal modo di gestire le passphrase anche se nella maggior parte delle applicazioni cryptsetup-luks è più conveniente.

Andrea Pasquinucci
pasquinucci@ucci.it

Riferimenti Bibliografici

[1] http://www.saout.de/misc/dm-crypt/, http://clemens.endorphin.org/LUKS

[2] Burt Kaliski, RFC 2898: PKCS #5 Password-Based Cryptography Specification

[3] http://www.freeotfe.org/

```
modprobe aes
modprobe sha256
modprobe dm_mod
modprobe dm_crypt

dd if=/dev/urandom of=/dev/hdd1

cryptsetup -v -y -c aes -s 256 luksFormat /dev/hdd1
cryptsetup luksOpen /dev/hdd1 dm-crypt-dev

mke2fs -j  /dev/mapper/dm-crypt-dev
mkdir /home/user/dm-crypt-dir
chown user.group /home/user/dm-crypt-dir
mount /dev/mapper/dm-crypt-dev /home/user/dm-crypt-dir
chown user.group /home/user/dm-crypt-dir

umount /home/user/dm-crypt-dir
cryptsetup luksClose dm-crypt-dev
```

Tabella 1. Creazione del filesystem cifrato

```
cryptsetup luksOpen /dev/hdd1 dm-crypt-dev
mount /dev/mapper/dm-crypt-dev /home/user/dm-crypt-dir
```

Tabella 2. Attivazione utente del filesystem cifrato

```
umount /home/user/dm-crypt-dir
cryptsetup luksClose dm-crypt-dev
```

Tabella 3. Disattivazione utente del filesystem cifrato

```
cryptsetup luksAddKey /dev/hdd1
cryptsetup luksDelKey /dev/hdd1 NUM
```

Tabella 4. Cambio password del filesystem cifrato

```
# Create a 100MB file named dm-crypt-file
DimMB="100"
dd if=/dev/urandom of=/home/user/dm-crypt-file bs=1M count=$DimMB
chown -R user.group /home/user/dm-crypt-file

# Attach the dm-crypt-file to the loop0 device
losetup /dev/loop0 /home/user/dm-crypt-file
```

```
# Detach the dm-crypt-file from the loop0 device
losetup -d /dev/loop0
```

Tabella 5. Creazione e gestione di un file che contiene un filesystem cifrato