Utenti e Permessi

Andrea Bontempi

POuL

Corsi Linux 2013

Multiutenza

- Uno dei cardini dell'informatica moderna è il multitasking, insieme alla possibilità di fare più cose "in contemporanea" si è però affiancata anche l'esigenza che più persone possano accedere ed usare un computer allo stesso stempo.
- La maggior parte dei sistemi operativi supporta la multiutenza, anche se le sue potenzialità non sono sempre sfruttate al massimo.
- La multiutenza comporta la gestione appunto di utenti ed eventualmente dei relativi permessi che ogni utente ha.

Multiutenza

- GNU/Linux gestisce nativamente più utenti ed i relativi permessi.
- Esiste un unico utente "amministratore" chiamato *ROOT*, in quanto proprietario della "radice" del filesystem, e quindi dell'intero sistema.
- Tutti gli altri utenti sono "classici" e possono acquisire temporaneamente o permanentemente dei permessi di amministrazione.
- Su ogni file possono essere assegnate delle regole di lettura, scrittura o esecuzione (poi vedremo come). Da notare che l'utente ROOT non subisce queste regole, e può accedere ad ogni file nel sistema (per questo va usato con cautela!)

/etc/passwd

La riga che identifica ROOT in /etc/passwd

root:x:0:0:root:/root:/bin/bash

Campi relativi ad un'utente:

1. Nome utente, solitamente tutto in minuscolo, senza spazi.
2. Ex-password, la 'x' indica che la password è salvata in /etc/shadow
3. UID - User ID (Il valore '0' è sempre riservato a ROOT, solitamente gli utenti normali hanno un ID superiore a 1000)
4. GID - Group ID (ID del gruppo "principale" a cui appartiene l'utente)
5. GECOS (Informazioni dell'utente. Come nome, cognome ...)
6. Home Directory, ovvero la cartella predefinita assegnata all'utente.
7. Shell di login, ovvero l'interprete dei comandi che l'utente potrà usare al login. Da notare che inserendo il comando "/sbin/nologin" inibiremo l'accesso a quel singolo utente. (Anche se ROOT può lo stesso usarlo)

useradd e usermod

Il comando useradd e usermod

useradd [OPZIONI] nomeutente
usermod [OPZIONI] nomeutente

- Se non si aggiungono opzioni, verranno usate le informazioni già esistenti oppure predefinite.
- *-d <path cartella>* Serve a specificare manualmente l'Home Directory. (*-m* la crea se non esiste)
- *-g <gruppo>* Serve a specificare quale è il gruppo principale dell'utente.
- -G <gruppi secondari> Aggiunge l'utente ai gruppi secondari. (separati da una virgola)
- -s <path shell> Specifica la shell predefinita per l'utente.
- Per rimuovere un'utente basta usare il comando "userdel nomeutente", aggiungendo l'opzione -r si rimuovono anche i file nella sua Home Directory.

/etc/shadow

- E' un file testuale strutturato come /etc/passwd, dato che contiene gli hash delle password può leggerlo soltanto l'utente ROOT.

La riga che identifica ROOT in /etc/shadow

root:6hashdellapassword:9999::::::

1. Nome utente
2. Hash della password. Il marcatore iniziale "6" indica l'algoritmo di hashing usato, nel nostro caso "SHA-512".
   - Il carattere '*' indica che il login è disabilitato.
   - Il carattere '!' vuol dire che non è ancora stata settata una password.
3. Data dell'ultima modifica della password dell'utente (Giorni trascorsi da *epoch*, ovvero il 1 gennaio 1970)
4. I campi successivi servono per gestire i limiti temporali, come ad esempio la scadenza della password.

/etc/group

- Il file è testuale e raccoglie tutti i gruppi esistenti nel sistema e gli utenti che ne fanno parte.

La riga che identifica il gruppo "wheel"

wheel:x:10:root,utente1,utente2

1. Nome del gruppo
2. Password del gruppo
3. GID - Group ID
4. Lista degli utenti appartenenti al gruppo, separati da una virgola.

passwd

- Questo comando serve a gestire le password dell'utente (per la precisione lavora su /etc/shadow)

Il comando passwd

passwd [OPZIONI] nomeutente

- Lanciando il comando senza opzioni parte il tool per cambiare la password dell'utente.
- Se non inserisci il nome utente viene preso l'utente di chi lancia il comando.
- Solo ROOT può cambiare la password di altri utenti.
- *-d* Serve a cancellare la password dell'utente.
- *-e* Serve a far scadere la password in modo forzato. Al primo login sarà richiesto di cambiarla.

Il comando su

su nomeutente [OPZIONI]

- Serve a cambiare l'utente loggato, previa richiesta della password dell'utente.
- L'utente ROOT può loggarsi con qualunque utente senza inserire alcuna password.
- *-c* Serve a eseguire un singolo comando con l'utente scelto, senza cambiare l'utente loggato. Per fare questa cosa è comunque suggertito l'uso di "sudo" (come vedremo successivamente).

I permessi standard

- Ogni file nel sistema è assegnato ad un utente (il proprietario) e ad un gruppo. Non è possibile per un file appartenere a più utenti o a più gruppi.
- Per ogni file è memorizzata nel filesystem una lista di permessi di lettura, scrittura ed esecuzione.
- Questi permessi sono separati per il proprietario, per il gruppo e per tutti gli altri. Possiamo quindi ad esempio avere un file con permessi di lettura/scrittura per il proprietario, con permesso di sola lettura agli utenti di un certo gruppo, e vietare l'accesso a tutti gli altri.
- Notare che anche le cartelle sono dei file e hanno questi permessi. L'unica differenza è che il permesso di esecuzione significa "attraversare". Ovvero impedisce l'accesso all'intero albero del filesystem a partire da quel nodo. (Infatti il permesso di lettura indica la vista del contenuto della cartella, ma se sai già il nome di un file contenuto puoi legtar -jxvf id2.tar.bz2 gerlo, sempre che si abbiano i diritti di lettura sul file)

I permessi speciali

- Sempre per ogni file esiste anche un gruppo di permessi "speciali" assegnabili per tutti gli utenti del sistema.
- *Sticky bit* - Indica che un file o una cartella (e quindi tutti i files contenuti) possono essere rimossi, rinominati o spostati solo dal proprietario del file (o cartella). E' utile nel caso in cui una cartella viene condivisa tra più utenti.
- *Setuid* - Indica che quell'eseguibile (se assegnato su altri tipi di file o viene ignorato o indica altri dettagli) può essere eseguito con i permessi assegnati al proprietario e non con quelli dell'utente che lo ha lanciato.
- *Setgid* - Come setuid, ma permette di avviare l'eseguibile con i permessi del gruppo a cui appartiene l'eseguibile.

chmod

- Questo comando serve per modificare i permessi di un file.

Il comando chmod

chmod [PERMESSI] [OPZIONI] nomefile

- I permessi possono essere scritti tramite notazione simbolica, preceduti da un + (per aggiungere il permesso) o da un - (per toglierlo).
- Con la notazione ottale è possibile aggiungere i permessi scrivendo un partiolare valore numerico a 4 cifre. La prima cifra indica i permessi speciali. La seconda i permessi per il proprietario, la terza i permessi per il gruppo e la quarta i permessi per tutti gli altri utenti.
- *-R* Serve a rendere ricorsivo il cambio dei permessi. Se fatto su una cartella segue l'albero del filesystem da quel punto e cambia i permessi a tutti i file che trova.

chmod (notazione simbolica)

Esempio: aggiungi permessi di lettura e scrittura a tutti

chmod a+rw nomefile

chmod (notazione simbolica)

Esempio: aggiungi permessi di lettura e scrittura a tutti

chmod a+rw nomefile

Permessi

- "r" - Lettura
- "w" - Scrittura
- "x" - Esecuzione
- *"s" - Setuid/Setgid*
- "t" - *Sticky bit*

Target

- *"u" - Proprietario del file*
- "g" - Gruppo del file
- "o" - Tutti gli altri®
- "a" - Tutti insieme

chmod (notazione ottale)

Esempio: aggiungi permessi di lettura e scrittura a tutti

chmod 0666 nomefile

chmod (notazione ottale)

Esempio: aggiungi permessi di lettura e scrittura a tutti

chmod 0666 nomefile

DA EVITARE

chmod 0777 -R /

chmod (notazione ottale)

Esempio: aggiungi permessi di lettura e scrittura a tutti

chmod 0666 nomefile

DA EVITARE

chmod 0777 -R /

Codici permessi standard

- 1 - Esecuzione
- 2 - Scrittura
- 4 - Lettura

Codici permessi speciali

- 1 - *Sticky bit*
- 2 - Setgid
- 4 - Getuid

chown

Il comando chown

chown nomeutente [OPZIONI] nomefile

- Permette di cambiare proprietario di un file.
- Il comando può essere usato soltanto dall'utente ROOT.
- *-R* Permette di effettuare il cambio in modo ricorsivo (esempio su una cartella e su tutti i files contenuti)

Il comando chgrp

chgrp nomeutente [OPZIONI] nomefile

- Permette di cambiare il gruppo di un file.
- Il comando può essere usato soltanto dall'utente ROOT.
- *-R* Permette di effettuare il cambio in modo ricorsivo (esempio su una cartella e su tutti i files contenuti)

Esercizio

- http://tinyurl.com/corsilinux1
- tar -jxvf id2.tar.bz2
- make // gcc id.c -o id

sudo

- E' un metodo più semplice, sicuro e selettivo per gli utenti normali di accedere ad alcuni permessi di ROOT usando la propria password.
- I permessi che un'utente può avere sono impostabili dal file di configurazione /etc/sudoers
- Ponendo il comando "sudo" prima di un'altro comando, quest'ultimo sarà eseguito con i permessi di ROOT.
- Per accedere ad una shell di root (invece che usare "su") è consigliato usare "sudo -s".
- Per lanciare un comando tramite un'altro utente usare "sudo -u nomeutente".

/etc/sudoers

- Per editare il file è consigliato l'uso del comando ad-hoc "visudo".

Aggiungere privilegi ad un utente

root ALL=(ALL) ALL

/etc/sudoers

- Per editare il file è consigliato l'uso del comando ad-hoc "visudo".

Aggiungere privilegi ad un utente

root ALL=(ALL) ALL

Aggiungere privilegi ad un gruppo di utenti

%wheel ALL=(ALL) ALL

/etc/sudoers

- Per editare il file è consigliato l'uso del comando ad-hoc "visudo".

Aggiungere privilegi ad un utente

root ALL=(ALL) ALL

Aggiungere privilegi ad un gruppo di utenti

%wheel ALL=(ALL) ALL

Aggiungere privilegi ad un gruppo molto fortunato

%lucky ALL=(ALL) NOPASSWD: ALL

Sintassi di /etc/sudoers

Righe di esempio

root ALL=(ALL) NOPASSWD: ALL
%foo ALL=(utente1, utente2) /bin/comando1, /bin/comando2

1. Nome utente, se preceduto da '%' è da interpretarsi come nome di un gruppo.
2. Host di provenienza. Se settato su ALL accetta comandi provenienti da ogni sorgente.
3. Utenti "impersonificabili". Se settato su ALL permette di diventare qualsiasi utente. Si possono settare più utenti separandoli con una virgola.
4. (Opzionale) NOPASSWD - Non chiede la password.
5. Comandi eseguibili attraverso il comando "sudo". ALL indica tutti, altrimenti specificare una lista di comandi separati da password.

Permessi avanzati, le capabilities.

- Un eseguibile ottiene i permessi di chi l'ha lanciato, tranne quando ha il Setuid abilitato, in questo caso gira con i permessi di ROOT.
- Lanciare un programma con tutti i permessi esistenti può essere un problema per la sicurezza del sistema.
- E' possibile concedere dei particolari permessi, detti "capabilities", ad un certo eseguibile, che potrà essere lanciato da un'utente normale con permessi limitati, tranne che potrà accedere ai singoli permessi concessi.
- Un esempio è il comando "ping". Per funzionare necessita di accedere ai raw socket, possibilità concessa solo a ROOT. E' possibile assegnargli la capabilities "cap_net_raw" in modo che tutti gli utenti possano lanciarlo senza dagli i permessi di ROOT.

Gestire le capabilities

getcap [OPZIONI] nomefile
setcap [OPZIONI] [CAPABILITIES] nomefile

GRAZIE