

Linux:
Controllo di accesso, gestione
utenti, gruppi, permessi

<http://www.lugcr.it>

Perché ?

Perché devo utilizzare un preciso nome utente?

1 : salvaguardia del sistema

Utilizzando il sistema come amministratore (root) si corre il rischio di danneggiare lo stesso

2 : sicurezza

Utilizzando il sistema come utente il rischio che il proprio sistema diventi preda di estranei è molto basso

3 : riservatezza

Utilizzando il sistema come utente i propri dati sono protetti da occhi indiscreti

Come ?

nome_utente :

permette al sistema di dedicarci risorse, di riconoscerci, di distinguerci da altri utenti

password :

permette al sistema di verificare che “siamo proprio noi”

permessi e proprietà :

sono attribuiti a file e cartelle, permettono a noi e al sistema di capire chi può operare su di essi e, soprattutto, come

Permessi sui file e sulle directory

Possono essere visti indifferentemente come caratteri o come valori ottali, ogni file e directory possiede 10 campi descrittivi del proprio stato (nella rappresentazione a caratteri), ne possiamo controllare il valore col comando `ls -l`. Il loro significato ed i valori possibili sono:

1. Tipo di file: **d** = directory, **l** = link, - = file, ...
2. Permesso di lettura al proprietario del file: **r** = permesso accordato, - = permesso negato
3. Permesso di scrittura al proprietario del file: **w** = permesso accordato, - = permesso negato
4. Permesso di esecuzione al proprietario del file: **x** = permesso accordato, - = permesso negato
5. Permesso di lettura al gruppo di utenti del proprietario del file: **r** = permesso accordato, - = permesso negato
6. Permesso di scrittura al gruppo di utenti del proprietario del file: **w** = permesso accordato, - = permesso negato
7. Permesso di esecuzione al gruppo di utenti del proprietario del file: **x** = permesso accordato, - = permesso negato
8. Permesso di lettura a tutti gli altri utenti: **r** = permesso accordato, - = permesso negato
9. Permesso di scrittura a tutti gli altri utenti: **w** = permesso accordato, - = permesso negato
10. Permesso di esecuzione a tutti gli altri utenti: **x** = permesso accordato, - = permesso negato

Permessi sui file e sulle directory

La rappresentazione ottale prevede di assegnare un valore numerico a ciascun tipo di permesso, così:

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

questi permessi possono essere attribuiti separatamente per:

- **Proprietario** del file (**user**)
- **Gruppo** a cui appartiene il proprietario (**group**)
- **Altri utenti** (**others**)

ad ognuno di questi “soggetti” può essere assegnata una combinazione dei tre permessi elementari, così se si vuole, ad esempio, indicare la possibilità di accedere in lettura e scrittura ad un file il valore ottale corrispondente sarà:

$$4 + 2 = 6$$

Permessi speciali

Sticky bit:

fa sì che il file possa essere cancellato esclusivamente dal proprietario, anche se altri hanno il permesso di scrittura

Gid:

attribuisce all'eseguibile in funzione i privilegi del gruppo a cui appartiene

Uid:

attribuisce all'eseguibile in funzione i privilegi dell'utente a cui appartiene

Permessi speciali

Esempio:

se creiamo un file con 'touch' esso sarà di “nostra” proprietà:

```
$ touch prova
```

```
$ ls -l prova
```

```
-rw-r--r--  1 matteo users      0 2004-03-18 15:02 prova
```

se attribuiamo a 'touch' lo Uid ogni nuovo file creato con touch avrà i permessi del proprietario di 'touch':

```
$ touch prova
```

```
$ ls -l prova
```

```
-rw-r--r--  1 root users      0 2004-03-18 15:06 prova
```

Come?

Come si editano i permessi?

`chmod` *[opzioni] (modo) opcode permesso file*

`chmod` *[opzioni] (modo) --reference=file_di_riferimento file*

modo (chi):

u = utente

g = gruppo

o = altri (others)

a = tutti (all)

opcode (come):

'+' = aggiunge un permesso

'-' = rimuove un permesso

'=' = assegna un permesso

modo è opzionale, se non specificato esso assume il valore di default:

a (all)

opcode funziona solo con la modalità testuale, con la modalità ottale si attribuiscono direttamente i permessi desiderati

Esempi

Alcuni esempi sui permessi:

`chmod u+x file`

aggiunge (+) il permesso di esecuzione (x) all'utente proprietario del file

`chmod 751 file`

imposta (modalità ottale) lettura, scrittura, esecuzione al proprietario (7 in prima posizione), lettura ed esecuzione agli altri utenti facenti parte del gruppo cui appartiene il proprietario (5 in seconda posizione), esecuzione a chiunque altro

`chmod u=rwx,g=rx,o=x file`

idem come sopra ma in modalità testo

`chmod =r file`

imposta (modo non specificato) lettura a tutti

`chmod 444 file`

imposta (modalità ottale) lettura a tutti

`chmod a-wx,a+r file`

per tutti gli utenti rimuove i permessi di scrittura ed esecuzione ed aggiunge solo quello di lettura

`chmod ugo=r`

imposta lettura a tutti

Impostazione di permessi speciali

Per impostare i permessi speciali:

- Uid = 4 oppure s
- Gid = 2 oppure s
- Sticky bit = 1 oppure t

esempi:

<code>chmod 4755 file</code>	assegna Uid
<code>chmod u+s file</code>	assegna Uid
<code>chmod 2755 file</code>	assegna Gid
<code>chmod g+s file</code>	assegna Gid

Proprietario

Come si modifica il proprietario di un file?

`chown` *[opzioni] nuovoproprietario file*

`chown` *[opzioni] --reference=file_di_riferimento file*

La umask

Domanda:

Come facciamo a far sì che ogni file creato da un utente abbia i permessi corretti?

Occorre “settare” la maschera dei permessi: **umask**

Questa maschera rappresenta i permessi che **NON** vengono attribuiti alla creazione del file, la mancata attribuzione dei permessi corrisponde alla sottrazione del loro codice ottale alla stringa ottale dei permessi.

PERMESSI DI DEFAULT – PERMESSI UMASK = PERMESSI TOTALI

solitamente la umask si utilizza col formato ottale si può, tuttavia, utilizzare anche il formato letterale, o simbolico, mediante il parametro:

-S

La umask

Esempio:

```
#umask 000          non modifico i permessi di default nel creare nuovi files
```

```
#touch prova
```

```
#ls -l prova
```

```
-rw-rw-rw-  1 root  root      0 2004-03-19 09:29 prova          cioè 666
```

```
#rm prova
```

```
#umask 022          modifico i permessi di default nel creare nuovi files non impostando  
la scrittura per il gruppo del proprietario e per gli altri
```

```
#touch prova
```

```
#ls -l prova
```

```
-rw-r--r--  1 root  root      0 2004-03-19 09:30 prova          cioè 644
```

infatti $666 - 022 \text{ (umask)} = 644$

File di configurazione e programmi

login	Permette l'accesso al sistema
su	Permette di operare con l'identità di un altro utente
/etc/passwd	Tabella delle caratteristiche salienti degli utenti
/etc/group	Tabella delle caratteristiche salienti dei gruppi
/etc/shadow	Tabella delle parole d'ordine quando non sono in /etc/passwd
/etc/motd	Messaggio di apertura o messaggio del giorno
/etc/securetty	Elenco dei terminali da cui è consentito l'accesso all'utente root
/var/spool/mail/*	Messaggi di posta elettronica degli utenti

Login

Il sistema (in realtà un ben preciso programma che si chiama login) controlla che esista un utente con il nome dato (username, non nome effettivo), e che la password corrisponda a quella registrata nel file /etc/passwd

A questo punto, avete una "identità": lo username, a cui corrisponde uno userid (o uid) numerico; e appartenente ad alcuni gruppi, a cui corrispondono dei groupid (o gid) numerici.

Login

Durante la fase di login diamo questi input:

Username

Password

se il login ha successo otteniamo questi output:

data e ora dell'ultimo accesso

eventuale presenza di posta non letta

motto del giorno

accesso

directory iniziale

Login

Per controllare l'ultimo accesso al sistema:

#lastlog

Per controllare chi attualmente è connesso:

\$users

\$who

\$w

Login

Per controllare la propria identità:

`$whoami`

Matteo

`$id`

`uid=1001(Matteo) gid=100(users) gruppi=100(users),6(disk),11(floppy)`

`$logname`

restituisce il nome dell'utente con cui si ha inizialmente avuto accesso al sistema

Utenti

Le informazioni sugli utenti sono contenute nel file:

`/etc/passwd`

il cui contenuto è organizzato come segue:

`utente:parola_d'ordine_cifrata:uid:gid:dati_personali:directory_home:shell`

<code>utente</code>	nome utente (username)
<code>parola_d'ordine_cifrata</code>	password cifrata, può invece essere contenuta in <code>/etc/shadow</code>
<code>uid</code>	User ID, identificativo utente (numerico)
<code>gid</code>	Group ID, identificativo gruppo (numerico)
<code>dati_personali</code>	descrittivo utente (solitamente nome e cognome)
<code>directory_home</code>	solitamente <code>/home/nomeutente</code> ma può variare
<code>shell</code>	shell predefinita dell'utente (solitamente <code>/bin/bash</code>)

Utenti

Storicamente il file `/etc/passwd` contiene l'elenco di tutti gli utenti e la loro password in forma criptata. Tutti gli utenti devono poter aver accesso in lettura a questo file, per cui l' esporre le password di tutti, seppur criptate, risulta rischioso per la sicurezza del sistema.

Si è, allora, affiancato al normale `/etc/passwd` il file `/etc/shadow` che introduce nuove funzionalità:

- permesso in lettura solo a root, viene lasciato l'accesso in lettura a `/etc/passwd` per tutti gli utenti
- le password in `/etc/shadow` sono criptate con algoritmi più complessi e robusti
- è possibile gestire il tempo di scadenza, la durata minima e massima e i tempi di notifica della password

Notare che:

- se la password è scritta in `/etc/shadow`, in `/etc/passwd` c'è solo una 'x' al posto della password criptata
- se in `/etc/passwd` il campo password è un '*', la password è nulla e l'utente non può accedere al sistema

Utenti

Contenuto di: `/etc/shadow`

Username:password:lastchange:min:max:warn:inactive:expire:

Username	Il nome dell'utente a cui fa riferimento la password
Password	Password criptata (13 caratteri). Può assumere anche altri valori quali * (asterisco) che sta ad indicare che l'utente è disabilitato e !! (o nessun carattere) che significa che l'utente non ha password (cosa molto pericolosa in termini di sicurezza)
lastchange	Numero di giorni compresi fra il 1 gennaio 1970 e l'ultima modifica della password
min	Minimo numero di giorni dall'ultima data di modifica prima di poter nuovamente cambiare la password
max	Durata massima della password (sempre in giorni)
warn	Numero di giorni di preavviso all'utente prima di invalidare la password
inactive	Numero di giorni di inattività possibili per quell'utente
expire	Data dopo la quale quel login non può più essere usato

Utenti

Le informazioni sui gruppi sono contenute nel file:

`/etc/group`

il cui contenuto è organizzato come segue:

`gruppo:parola_d'ordine_cifrata:gid:lista_di_utenti`

`gruppo`

nome del gruppo

`parola_d'ordine_cifrata`

password del gruppo, di solito non si usa

`gid`

Group ID, identificativo numerico del gruppo

`lista_di_utenti`

lista di utenti appartenenti al gruppo, separati da ','

Utenti

Cambiare identità:

`$su:`

`su` per diventare root

`su nomeutente` per cambiare utente

`su -` per ri-inizializzare l'ambiente a quello del nuovo utente, eseguendo per intera la sequenza di login

Utenti

Gestione utenti:

<code>useradd</code>	Aggiunge un utente e tutto quello che serve perchè possa accedere
<code>userdel</code>	Cancella un utente
<code>usermod</code>	Modifica le informazioni dell'account
<code>/etc/skel/</code>	Contiene i file da aggiungere nella home di un nuovo utente
<code>passwd</code>	Permette di modificare la parola d'ordine
<code>chsh</code>	Cambia la shell abbinata all'utente
<code>/etc/shells</code>	Elenco delle shell utilizzabili nel sistema
<code>chfn</code>	Modifica i dati personali dell'utente

Utenti

useradd [opzioni] nomeutente

opzioni:

- d directory_home
- g gruppo_iniziale
- G gruppi_supplementari
- m (crea directory, se non esiste, usando /etc/skel)
- s shell

userdel [opzioni] nomeutente

opzioni:

- r rimuove la directory_home e tutti i file contenuti al suo interno

Utenti

usermod [opzioni] nomeutente

opzioni:

-c commento

imposta il campo “comment”

-d directory_home

imposta la directory_home

-e data di scadenza

imposta la data di scadenza dell'account

-f n°-giorni

disabilita l'account dopo che sono trascorsi n°-giorni dalla scadenza della password

-g gruppo

imposta il gruppo iniziale

-G gruppi

imposta i gruppi supplementari (lista senza spazi separata da virgole)

-l nome

imposta il nome di login

-s shell

imposta la shell di login

Utenti

Come modificare la password:

```
#passwd nomeutente
```

utilizzabile in questa forma solo da root: permette di cambiare la password ad altri utenti senza chiedere la vecchia password.

```
$passwd
```

permette di modificare la propria password, per sicurezza viene chiesta la vecchia password.

Si sconsiglia l'uso di accentate, potrebbe essere impossibile digitarle da alcuni terminali.

Utenti

Vi sono alcuni utenti particolari, non esistenti nella realtà, ai cui è demandato il “buon funzionamento” del sistema:

bin:x:1:1:bin:/bin:

daemon:x:2:2:daemon:/sbin:

adm:x:3:4:adm:/var/log:

lp:x:4:7:lp:/var/spool/lpd:

sync:x:5:0:sync:/sbin:/bin/sync

shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown

halt:x:7:0:halt:/sbin:/sbin/halt

mysql:x:27:27:MySQL:/var/lib/mysql:/bin/bash

rpc:x:32:32:RPC portmap user::/bin/false

sshd:x:33:33:sshd:/:

gdm:x:42:42:GDM:/var/state/gdm:/bin/bash

pop:x:90:90:POP:/:

nobody:x:99:99:nobody:/:

Gruppi

Le informazioni inerenti i gruppi sono raccolte nel file: /etc/groups

gruppo:parola_d'ordine_cifrata:gid:lista_di_utenti

gruppo

nome del gruppo

parola_d'ordine_cifrata

password del gruppo, di solito non si usa

gid

Group ID, identificativo numerico del gruppo

lista_di_utenti

lista di utenti appartenenti al gruppo, separati da ','

Gruppi

groupadd [opzioni] nomegruppo

opzioni:

-g gid

imposta il numero gid del gruppo

groupdel [opzioni] nomegruppo

groupmod [opzioni] nomegruppo

opzioni:

-g gid

imposta il numero gid del gruppo

-n nome

imposta il numero nome del gruppo

“Trucchi”

Utente con funzione specifica:

Creare un utente fittizio, con o senza parola d'ordine, al quale si associa un programma o uno script, al posto di una shell.

La directory corrente nel momento in cui il programma o lo script viene eseguito è quella indicata come directory home (directory personale).

L'esempio seguente mostra un record del file `/etc/passwd` preparato in modo da permettere a chiunque di eseguire il programma (o lo script) `/usr/local/bin/ciao` partendo dalla posizione della directory `/tmp/` (il numero UID 505 e GID 100 sono solo un esempio):

```
ciao::505:100:Ciao a tutti:/tmp:/usr/local/bin/ciao
```

“Trucchi”

Gruppo di utenti con lo stesso numero UID:

ogni utente avrebbe un proprio nome e una parola d'ordine per accedere al sistema, ma poi, tutti i file apparterrebbero a un utente immaginario che rappresenta tutto il gruppo.

Segue un esempio del file `/etc/passwd`:

```
tutti*:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
```

```
alfa:34gdf6r123455:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
```

```
bravo:e445gsdfr2124:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
```

```
charlie:t654df7u72341:1000:1000:Gruppo di lavoro:/home/tutti:/bin/sh
```

Se esiste la necessità o l'utilità si possono assegnare anche directory personali e shell differenti.

“Trucchi”

Un gruppo per ogni utente:

ogni volta che si crea un nuovo utente, si crea anche un gruppo con lo stesso nome e, possibilmente, lo stesso numero (UID = GID). Questa tecnica si combina con una maschera dei permessi 002.

I file vengono creati in modo predefinito con i permessi di lettura e scrittura, sia per l'utente proprietario che per il gruppo, mentre si esclude la scrittura per gli altri utenti, in questo modo si può concedere a un altro utente di poter partecipare al proprio lavoro: basta aggiungere il suo nome nell'elenco degli utenti associati al proprio gruppo.

Si possono creare degli altri gruppi aggiuntivi, in base alle attività comuni e aggiungere a questi gruppi i nomi degli utenti che di volta in volta partecipano a quelle attività. Naturalmente, i file da condividere all'interno dei gruppi devono appartenere a questi stessi gruppi.

“Trucchi” - esempio

Vediamo, per esempio, cosa sia necessario fare per gestire un gruppo di lavoro per un ipotetico progetto «alfa»:

- 1 la maschera dei permessi predefiniti (umask) degli utenti che faranno parte del progetto sia pari a 002, per consentire ogni tipo di accesso ai file e alle directory che verranno create da parte degli utenti del gruppo del progetto.
- 2 Si crea il gruppo alfa e a questo si abbinano tutti gli utenti che dovranno fare parte del progetto. Il record del file `/etc/group` potrebbe essere simile a quello seguente:
alfa::101:tizio,caio,sempronio
- 3 Si crea una sorta di directory home per i file del progetto, con eventuali ramificazioni:
mkdir /home/progetti/alfa
mkdir /home/progetti/alfa/...

“Trucchi” - esempio

4 Si assegna l'appartenenza di questa directory (ed eventuali sottodirectory) al gruppo di lavoro.
`# chown -R root.alfa /home/gruppi/alfa`

5 Ciò che viene creato all'interno del gruppo di directory deve appartenere al gruppo delle directory stesse:

```
# chmod -R 2775 /home/progetti/alfa
```

tutte le directory del progetto ottengono l'attivazione del bit GID, attraverso il quale, in modo predefinito, i file creati al loro interno apparterranno al gruppo delle directory stesse.

Ringraziamenti

Intendo ringraziare:

Gianluca Storti

Paolo Poletti

Daniele Giacomini

Ellen Siever, Stephen Spainhour, Stephen Figgins, Jessica P. Hekman

dai cui documenti ed opere ho tratto questi lucidi

MATTEO VICINI