Autodir HOWTO

Venkata Ramana Enaganti

ramana <> intraperson dot com

23-9-2004

Diario delle Revisioni

Revisione 1.04 25-5-2007 Revisionato da: VRE
Piccoli aggiornamenti
Revisione 1.03 15-9-2006 Revisionato da: GaMA
Revisione richiesta dall'autore.
Revisione 1.02 25-12-2004 Revisionato da: VRE
Piccoli aggiornamenti
Revisione 1.00 23-9-2004 Revisionato da: VRE
Prima versione, rivista da Rahul Sundaram di TLDP
Revisione 0.32 13-9-2004 Revisionato da: VRE
Nuove sezioni come i requisiti ed altro.
Revisione 0.10 24-6-2004 Revisionato da: VRE
seconda bozza
Revisione 0.9 11-6-2004 Revisionato da: VRE
Prima bozza

Questo HOWTO è dedicato all'installazione e configurazione di **Autodir**, e ad altre problematiche inerenti **Autodir**.**Autodir** è spesso utilizzato per permettere un facile accesso alle directory home. Traduzione a cura di Gualtiero Testa (gualty at libero.it). Revisione a cura di Elisabetta Galli (lab at kkk.it)

1. Introduzione

Autodir offre un metodo semplice ed efficace per creare, in maniera trasparente, directory come le directory home. Autodir si appoggia al protocollo autofs (htp://www.autofs.org) per il suo funzionamento.

Questo documento spiega come creare directory su richiesta usando **Autodir** in maniera trasparente alle applicazioni. Il documento spiega inoltre il sistema di backup trasparente che è possibile con **Autodir**, senza una interruzione per backup del sistema; questa soluzione si applica a tutte le directory gestite da **Autodir**.

1.1. Licenza e Copyright

1.2. Liberatoria

L'autore non si assume alcuna responsabilità per il contenuto di questo documento. Usare i concetti, gli esempi e le informazioni a proprio rischio e pericolo. Ci possono essere errori ed imprecisioni che potrebbero danneggiare il sistema. Procedere con cautela e, sebbene un danno sia altamente improbabile, l'autore non si assume alcuna responsabilità.

Tutti i marchi registrati appartengono ai rispettivi proprietari, a meno che non sia indicato diversamente. L'uso di un termine in questo documento non deve essere considerata violazione della validità del marchio. La citazione di specifici prodotti o marchi non deve essere considerata come approvazione.

1.3. Commenti

Commenti su questo documento sono sicuramente benvenuti. Inviare aggiunte, commenti e critiche al seguente indirizzo di email:<`ramana <> intraperson dot com`>.

1.4. Nuove versioni di questo documento

L'ultima versione di questo HOWTO è disponibile su http://www.intraperson.com/autodir/.

1.5. Crediti / contributi

In questo documento, ho il piacere di evidenziare per la revisione tecnica e linguistica:

- Rahul Sundaram<`rahulsundaram@yahoo.co.in`>
- Machtelt Garrels

2. Prima di entrare nei dettagli...

Dopo aver rilasciato la versione beta di intraperson, ho iniziato a lavorare sulla guida di amministrazione che tratta gli aspetti di amministrazione di **intraPerson**. Per maggiori dettagli controllare su http://www.intraperson.com. Ma sono stato bloccato da una semplice cosa. E' facile creare utenti in LDAP, o almeno lo penso io, ma come creare le directory home per quegli utenti in LDAP indipendentemente da dove gli account sono importati ?

Ho trovato alcune soluzioni ma non sono soddisfatto perché ognuna aveva dei seri inconvenienti. Dopo aver sfogliato la documentazione di Autofs e aver fatto un po' di prove, sono arrivato alla conclusione che il protocollo Autofs può offrire una soluzione migliore a questa sfida.

Il risultato è **Autodir**, basato sul protocollo Autofs.

3. Perché non il pam_mkhomedir?

Il modulo PAM `pam_mkhomedir` utilizza l'architettura Pluggable Authentication Module per questa operazione. Così com'è, ci sono delle limitazioni. Per esempio:

- Alcuni server potrebbero non autenticare gli utenti ma aspettarsi che le loro directory esistano. Questo significa che non utilizzano PAM e, a sua volta, che `pam_mkhomedir` non ha la possibilità di creare le directory home. L'esempio più tipico è quello dei server di posta.
- PAM è sempre un componente opzionale per l'autenticazione. Alcuni servizi potrebbero non usare PAM ed utilizzare un diverso metodo di autenticazione degli utenti. In questo caso `pam_mkhomedir` non sarà mai usato.
- In genere `/home` è controllata da root e solo gli utenti root possono creare directory home al suo interno. Dunque il servizio che vuole creare directory home con PAM deve essere eseguito come root oppure la directory home deve avere gli stessi permessi come, per esempio, `/tmp`.

Infine **Autodir** ha un utilizzo più ampio e supporta funzionalità più interessanti.

4. Dove può essere usato Autodir ?

- Dove gli account utente risiedono in un database centralizzato come LDAP, SQL, NIS, NIS+ o altri database, dai quali utenti e gruppi sono importati su altri sistemi. Per creare, ad esempio in home, directory di gruppi su quei sistemi che importano gli account da un database centralizzato, su richiesta.
- Per sfruttare la sua funzionalità di backup trasparente per i sistemi in linea 24 ore su 24.
- Può anche essere utilizzato quando gli account sono sul sistema locale, nascondendo per certi versi quali account esistono nella directory `/home`, per esempio.

5. Cosa non è Autodir

Autodir può creare directory ma non può rimuoverle quando gli utenti e/o gruppi sono rimossi dal database di account. Per farlo utilizzare script di cron personalizzati.

6. Differenze tra Autodir e Autofs

Potrebbero esserci problemi nel caso si utilizzi già il pacchetto Autofs per gestire il montaggio delle directory home. Analizziamo le differenze tra i due pacchetti:

- L'obiettivo principale di autofs è quello di gestire montaggi di rete a richiesta invece di montarli tutti allo stesso tempo, il che preserva le risorse di sistema. Sebbene ci sia, nel pacchetto autofs, un supporto per il montaggio dinamico di directory home, il requisito è che *queste directory home esistano già*.

  D'altro lato, **Autodir** è specializzato *solo* nella creazione e montaggio di directory locali.

  Autodir può anche creare directory reali su file system su disco le quali non risiedono in una singola directory base. Questo è quello che fanno per impostazione predefinita utility come **useradd**. In un file system standard, tutte le directory home risiedono nella directory base `/home`. Per file system come ext2 e ext3, le prestazioni degradano se un numero elevato di directory home esiste in una singola directory base.

  Per le applicazioni che accedono a queste directory, **Autodir** le presenta *su richiesta* in una *singola* directory base virtuale; le effettive directory sono create in sottodirectory in qualche altra directory con approccio gerarchico.

  Per esempio, la home reale per l'utente di nome `user1` sarà creata in `/autohome/u/us/user1` se configurata in questo modo, ma montata in `/home` su richiesta delle applicazioni che accedono alla directory home come `/home/user1`.

  Le autorizzazioni per la directory base reale, dove le directory home sono effettivamente create (`/autohome` nell'esempio sopra), sono gestite in modo che solo root possa accedere a `/autohome`.

  Il montaggio delle directory su richiesta, e lo smontaggio quando non più in uso, porta ad una interessante opportunità: poter dire quando una directory è in uso o no. Se la directory non è in uso, si può far partire senza pericoli un programma come una applicazione di backup, quando una directory non è montata.

  Autodir sfrutta questa possibilità facendo partire il menzionato comando di backup quando una directory diventa inutilizzata.

- C'è una ulteriore questione da illustrare per gli amministratori che leggono questo documento. **Autodir** non chiama i programmi esterni **mount** e **umount**, come nel caso del pacchetto autofs; usa invece direttamente chiamate di sistema. Come effetto secondario, Autodir è più veloce ed affidabile ma il file `mtab` non viene aggiornato. Ho ritenuto che non fosse necessario perché tutti i montaggio e smontaggi sono effettuati su directory locali.
- Un'altra piccola differenza è che **Autodir** è completamente *multi-threaded*. Anche autofs diventerà multi-threaded nelle versioni future.

7. Il funzionamento

Autodir utilizza i moduli per aggiungere funzionalità specifiche. **Autodir** implementa funzionalità generiche che i moduli sfruttano e su cui aggiungono funzioni specifiche.

Si può aggiungere solo un modulo alla volta a **Autodir**. Se ci sono due moduli, ad esempio `autohome` e `autogroup`, allora occorre creare due processi di **Autodir** in modo che ogni processo abbia un solo modulo agganciato.

Per le successive spiegazioni, utilizzerò il modulo `autohome` che gestisce in maniera trasparente la creazione delle directory home.

- Il modulo `autohome` crea le directory home degli utenti su richiesta se queste non esistono già.
- Si assume che gli account utenti esistano ma non le relative directory home, o perché questi account sono stati creati con l'opzione `-M` di **useradd** o perché sono stati importati da LDAP, NIS o altri database esterni per i quali le directory home devono essere ancora create.
- Si assume anche, *ma solo per questa spiegazione*, che tutte le directory home degli utenti debbano essere nella directory `/home`.

KISS: Keep it Simple: alcuni dettagli di fino sono stati intenzionalmente tralasciati per rendere la spiegazione facile da capire.

Prima di tutto il file system autofs è montato sulla directory `/home` da **Autodir**. Il kernel di Linux viene informato che `/home` è gestita d'ora in avanti da una applicazione in user space, **Autodir**.

Autofs ?: Non ci si deve preoccupare troppo del file system autofs se non lo si capisce. E' sufficiente considerarlo un file system speciale, simile ai file system in memoria ma con alcune proprietà speciali.

```
+----------------+
```

```
                              |  Linux Kernel  |
                              +----------------+
                                /        \
                               /          \
                              /            \
                             /              \
+-------------+       +---------+       +------------+       +-----------------++----
| Application |------>| /home   |<----->|  Autodir   |<------>| autohome module |
+-------------+       +---------+       +------------+       +-----------------++----
                           \                                           /
                            \  +----------------+                     /
                             +-|   /autohome    |<-------------------+
                               +----------------+
```

Quando una applicazione o un demone richiede l'accesso alla directory home di un utente, per esempio `/home/userhome1`, entrano direttamente in `/home/userhome1` per accedervi. Il kernel rileva l'accesso ed informa **Autodir** se la directory `userhome1` non esiste ancora in `/home`.

Autodir, a sua volta, passa questa richiesta al modulo `autohome`. Il modulo `autohome` non tocca la directory `/home`. Invece gestisce le *vere directory home* da qualche altra parte, per esempio in `/autohome` come mostrato in figura.

Il modulo `autohome` crea una directory home reale se non esiste nella directory `/autohome`. Dopo la creazione o il fallimento del tentativo, qualunque sia il risultato, `autohome` informa **Autodir**. Quando la creazione della directory è stata completata con successo, il percorso alla directory home reale è fornito **Autodir**.

Se il modulo `autohome` riporta successo, **Autodir** crea la directory `userhome1` sotto `/home` e monta la *directory home reale* da `/autohome` su di essa. Alla fine del processo, **Autodir** informa il kernel se l'intera operazione ha avuto successo o no. Di conseguenza, il kernel permette alle applicazioni di entrare nella directory o, in caso di fallimento, riporta che la directory non esiste.

8. Alcune definizioni

Prima di procedere oltre e per semplificare la spiegazione, è meglio capire i seguenti termini.

Directory virtuali

Queste directory non esistono su disco. Sono invece create e cancellate su richiesta in memoria. Se il sistema riparte tutte queste directory svaniscono. Nella precedente figura, tutte le directory sotto `/home` sono *directory virtuali*.

Directory base virtuale

Questa è la directory che contiene tutte le *directory virtuali*. Questa directory *esiste* su disco e quindi rimane anche dopo un riavvio. Nella figura precedente, `/home` è *la directory base virtuale*.

Directory reali

Queste sono le directory che risiedono effettivamente su disco. Anche dopo un riavvio, queste rimangono intatte. Nella precedente figura, tutte le directory create in `/autohome` sono *directory reali*.

Directory base reale

è la directory che contiene tutte le directory reali. Nella figura precedente, `/autohome` è *la directory base reale*.

Ogni *directory virtuale* è mappata su una *directory reale*. Questo significa che quello che è scritto o modificato nella *directory virtuale* è in realtà mandato nella *directory reale*.

Al riavvio del sistema le *directory reali* ed il loro contenuto rimangono intatte. Ma le *directory virtuali* sono di nuovo create su richiesta, esattamente come prima.

Le *directory virtuali* sono rimosse se non utilizzate per un determinato periodo di tempo e create di nuovo, se necessario. Quando una *directory virtuale* è rimossa, il programma di backup, se configurato, verrà avviato sulla corrispondente *directory reale*.

> **Importante:** Le applicazioni dovrebbero accedere solo alle *directory virtuali*. Le *directory reali* sono nascoste alle applicazioni. Solo l'utente root può vederle. Esiste una sola eccezione: il programma di backup può sempre accedere alle sole *directory reali*.

9. Organizzazione delle directory nella directory base reale

Perché ci dovrebbe essere una particolare organizzazione delle directory nella *directory base reale*? Se tutte le *directory reali* vengono create in un unica *directory base reale* ci potrebbe essere un decadimento delle prestazioni a causa dell'alto numero di *directory reali* da creare. File system come ext2 e ext3 non sono ottimizzati per questo tipo di struttura piatta di directory.

Sarebbe molto meglio se la *directory base reale* fosse divisa in molte sottodirectory, o anche se queste sottodirectory fossero divise a loro volta in altre sottodirectory. Le effettive directory home sarebbe collocate nell'ultima sottodirectory.

Ci sono tre tipi di organizzazione per la directory:

Livello 0

Di fatto nessuna organizzazione. Tutte le directory home sono create direttamente nella *directory base reale*.

Livello 1

La *directory base reale* è divisa in più sottodirectory. I nomi delle sottodirectory sono derivati dal primo carattere della directory finale da creare. Per esempio, se deve essere creata la directory `user1`, viene prima creata la directory di nome 'u' sotto la *directory base reale*. Quindi in quella sottodirectory l'effettiva directory `user1` viene creata come `/<directory_base_reale>/u/user1`.

Livello 2

Come l'organizzazione di livello 1, ma dopo il primo livello di sottodirectory ne viene creato un secondo. I nomi sono i primi due caratteri del nome della directory finale da creare. Per esempio, per l'utente `user1`, come nell'esempio precedente, viene creata la directory `/<directory_base_reale>/u/us/user1`.

10. La scadenza della directory virtuale

Quando una applicazione cerca di accedere ad una *directory virtuale* in una *directory base virtuale*, **Autodir** crea la *directory virtuale* se non esiste già e monta su di essa la *directory reale* dalla *directory base reale*. Una volta terminato, viene fatto partire un timer. Se non ci sono accessi alla *directory virtuale* nella *directory base virtuale* da parte di qualsiasi applicazione per un specificato periodo di tempo, la directory viene rimossa e la corrispondente *directory reale* nella *directory base reale* è marcata per il backup.

Il periodo di tempo di scadenza può essere specificato attraverso un' opzione della linea di comando di **Autodir**.

11. Supporto per il backup

Autodir supporta il lancio di programmi di backup quando una *directory virtuale* è rimossa dopo un periodo di attività La rimozione della *directory virtuale* è di per se una garanzia che nessuna applicazione possa accedervi e modificarla.

Così come c'è un tempo di attesa per la scadenza di una *directory virtuale*, anche per il backup **Autodir** attende un periodo di tempo dopo la scadenza della *directory virtuale*, prima di far partire il backup. Questo periodo di tempo può essere specificato attraverso un' opzione della linea di comando di **Autodir**.

Per impostazione del progetto, i programmi di backup operano sulle *directory reali* ma non sulle *directory virtuali*. Se il programma di backup cerca di accedere ad una *directory virtuale*, allora **Autodir** suppone che una applicazione normale abbia bisogno di quella directory e quindi termina il programma di backup, anche se il processo che sta accedendo alla *directory virtuale* è proprio il programma di backup.

Viene creato un processo di backup separato per ogni *directory reale*. Al programma di backup possono essere fornite come argomento le *directory reali* sulle quali operare.

> **Nota:** Il supporto per il backup è indipendente dal modulo usato. E' applicabile a tutti i moduli di **Autodir**.

> **Backup è come dire reale!:** I programmi di backup non dovrebbero mai accedere alla *directory virtuale* o alla *directory base virtuale*.

> **Avvertenza**
>
> La funzionalità di backup non è molto utile se le *directory virtuali* sono sempre utilizzate dalle applicazioni.

12. Requisiti per il programma di backup

Autodir richiede alcuni requisiti extra al programma di backup utilizzato: quando il backup è in corso sulla *directory reale*, con la corrispondente *directory virtuale* scaduta, e quella *directory virtuale* è richiesta di nuovo da una applicazione, il processo di backup è terminato. Prima di tutto viene mandato un segnale di `SIGTERM` per fermarlo in maniera gentile. Ma se non si ferma in tempo, ed ha un secondo per farlo, viene mandato un segnale di `SIGKILL` , il quale sicuramente lo fermerà.

> **Nota:** Solo quando il backup è stato fermato, l'applicazione può accedere alla **directory virtuale** richiesta.

> **Importante:** Qualunque programma di backup sia usato, deve essere in grado di recuperare da questo segnale in maniera corretta, senza causare effetti secondari irrimediabili.

Un altro aspetto importante è l'ambiente nel quale il backup viene eseguito. Tutti i programmi di backup sono eseguiti come root. Ma allo stesso tempo tutti i privilegi di root non necessari sono rimossi utilizzando le funzionalità POSIX. In altre parole questi programmi di backup possono leggere qualsiasi file o directory che appartiene agli utenti del sistema e niente di più. A parte questo, il processo di backup si comporta come un normale processo a livello utente.

13. Opzioni dei moduli

Ci sono due tipi di opzioni che si possono passare ad **Autodir**. Quelle del primo tipo sono opzioni per **autodir** stesso e sono comuni a qualsiasi modulo venga utilizzato. Lo opzione dell'altro tipo sono specifiche del modulo usato. Queste sono dette sotto opzioni e sono passate al modulo utilizzato; sono distinte da quelle principali dall'opzione `-o`. Il tutto è molto simile alle sotto opzioni utilizzate con il comando **mount**.

Per esempio, le sotto opzioni per il modulo di esempio `autohome` possono essere passate come segue:

```
-o 'realpath=/tmp/autohome,level=2,noskel'
```

Qui `realpath`, `level` e `noskel` sono sotto opzioni del modulo `autohome`.

14. Requisiti per Autodir

- Serve un kernel Linux in versione 2.4 o superiore. Queste versioni del kernel supportano il montaggio di una directory su un'altra. Al momento **Autodir** non è stato portato su altri tipi di Unix ma questo potrebbe cambiare in futuro.
- **Autodir** richiede che il modulo kernel autofs sia basato sul protocollo 4. Ma il pacchetto a livello utente autofs non è necessario. Il modulo kernel autofs è alquanto standard e praticamente tutte le distribuzioni lo includono.

15. Il modulo kernel autofs

Autodir utilizza il modulo kernel autofs per il suo funzionamento. Il modulo kernel `autofs` deve essere caricato prima di far partire **autodir**.

Si può fare da root, utilizzando il comando **modprobe** come segue:

```
# modprobe autofs
```

16. Importare gli account di utente e di gruppo

Se gli account di utente e di gruppo risiedono in un database centralizzato, devono essere importati prima di far partire **Autodir**. Come farlo è fuori dall'ambito di questo HowTo. Ci sono numerosi documenti che spiegano in maniera chiara come farlo.

17. Ottenere Autodir

Al momento **Autodir** è disponibile nei formati tar e rpm. Maggiori informazioni possono essere trovate su http://www.intraperson.com/autodir/ (http://www.intraperson.com/autodir/).

Dopo aver scaricato il codice sorgente, seguire i seguenti semplici passi:

- Scompattare il codice sorgente

```
$ tar zxvf <tar file name>
```

- Spostarsi nella directory espansa ed eseguire i seguenti comandi:

```
$ ./configure
```

```
$ make
```

```
# make install
```

> **Non funziona ?:** Lo script `configure` controlla la presenza delle librerie richieste. Se non ci sono si ferma.

18. Gestire le directory home

Questa sezione spiega come configurare **Autodir** in modo che le directory home degli utenti siano create su richiesta. Per questo scopo viene usato il modulo `autohome`, che si occupa in dettaglio della creazione delle directory home.

Per caricare il modulo `autohome` di **Autodir**, occorre usare l'opzione `-m`. Per esempio, `-m /usr/lib/autodir/autohome.so`.

Associazione utente/home: Quando una applicazione cerca di accedere ad una home directory, quella home directory è utilizzata per controllare se esiste un utente con lo stesso nome della directory. Se il nome dell'utente esiste allora la home directory è creata. Altrimenti viene restituito all'applicazione il messaggio “no such file or directory”.

Account di utente: `Autohome` non si occupa della creazione degli account nel sistema locale, in LDAP o in qualsiasi altro database. Gestisce solo la creazione delle directory home quando questi account esistono e sono importati nel sistema locale da database come LDAP e NIS.

Limitazioni: Vale la pena di citare una limitazione del modulo `autohome`. Il modulo assume che il nome dell'utente e la sua home directory siano collegati l'uno con l'altra. Per esempio, per l'utente `user1` la home directory dovrebbe essere `/home/user1` o `/una/directory/name/user1` ma non `/una/directory/nome/userhome1`. Questa situazione potrebbe essere gestita ma causerebbe un eccessivo uso delle risorse di sistema in quanto le righe del file password dovrebbero essere controllate dalla prima all'ultima.

Capire quando non usare autohome: Se il database degli utenti è definito in modo tale da distribuire le directory home tra diverse directory base, per esempio `/home/class1/user1`, `/home/class2/user2332`, allora la configurazione di `autohome` diventa troppo complicata e quindi il suo uso non raccomandato.

18.1. Directory base per autohome

Il passo successivo nella procedura di impostazione è decidere dove saranno create la *directory base virtuale* e la *directory base reale* per la creazione delle directory home.

Come sono definite la *directory base virtuale* e la *directory base reale* nel contesto del modulo `autohome`?

Dipende tutto da come gli account utente sono creati. Se per l'utente user1 viene creato un account utente con `/home/user1` come home directory allora `/home` *diventerà la directory base virtuale*.

Allora qual'è la *directory base reale*? Può essere qualsiasi directory. L'unica cosa da tenere a mente è verificare che ci sia abbastanza spazio perché tutti i file saranno salvati qui e non nella *directory base virtuale*.

Nella maggior parte delle configurazioni server `/home` è in una partizione separata. Ma se `/home` è la *directory base virtuale*, allora i file non sono memorizzati in quella directory! La soluzione è di non fare il montaggio della partizione su `/home` ma invece da qualche altra parte e renderla la *directory base reale*.

L'opzione di **Autodir** `-d` è utilizzata per specificare la *directory base virtuale*. Per esempio: `autodir -d /home` definisce `/home` come la *directory base virtuale*.

E' talvolta complicato specificare la *directory base reale*. La *directory base reale* è gestita dal modulo `autohome` quindi l'opzione deve essere passata al modulo attraverso le sotto opzioni di modulo. Se la *directory base reale* è `/var/autohome` allora è specificata con l'opzione `-o` come in `-o realpath=/var/autohome`.

18.2. Organizzazione delle directory

Si faccia riferimento alla organizzazione delle directory nella directory base reale per una dettagliata spiegazione dell'argomento.

`autohome` supporta questo tipo di organizzazione. La sotto opzione usata per specificare il tipo di organizzazione desiderato è l'opzione `level` per esempio: `-o level=2`.

18.3. Sotto opzioni varie per autohome

La sotto opzione `skel` può essere utilizzata se il percorso dei file di scheletro non è quello standard `/etc/skel`: `-o skel=/un/altra/dir`.

La sotto opzione `noskel` può essere usata con `-o` per indicare che i file di scheletro non devono essere copiati delle directory home quando sono create.

18.4. Esempio

Primo, importare gli account utente da un database centralizzato, come ad esempio LDAP.

Poi, il modulo kernel `autofs` deve essere caricato nel modulo Linux. Questo può essere fatto come descritto nella sezione sul modulo kernel autofs.

Se `/home` deve essere usato per le directory home allora `/home` diventa la *directory virtuale*; questo è indicato ad **autodir** con l'opzione `-d /home`.

Supponendo che il modulo `autohome` sia in `/usr/lib/autodir/autohome.so`, questo modulo può essere caricato insieme ad **autodir** con `-m /usr/lib/autodir/autohome.so`. Da notare che viene indicato il percorso completo.

La vera posizione delle directory home è indicata con la sotto opzione `realpath`. Se è `/autohome`, la posizione può essere specificata con `realpath=/autohome`.

Con tutte queste opzioni **autodir** può essere fatto partire con:

```
# autodir -d /home        \
  -m /usr/lib/autodir/autohome.so   \
  -o 'realpath=/autohome'      \
```

Una volta partito **Autodir**, la directory `/home` sarà inizialmente vuota. Per verificare se **Autodir** sta lavorando correttamente, si può entrare in una delle directory home come utente root o come il proprietario della directory.

19. Gestire le directory di gruppo

Il modulo `autogroup` può essere utilizzato per creare su richiesta directory di gruppo per un accesso comune. Può essere utilizzato con Samba, ad esempio, per creare dinamicamente directory condivise per un gruppo di persone.

> **Controlli:** Il modulo `autogroup` controlla che la directory richiesta sia per un gruppo valido all'interno del database di sistema dei gruppi.

> **Usare autogroup per la creazione delle directory home:** Il modulo `autogroup` può essere anche usato per creare le directory home, assumendo che esista un gruppo privato per ogni utente . In questo modo tutte le directory home e di gruppo possono essere create in un solo punto e con un solo modulo. Tuttavia, i file di scheletro non vengono copiati e la sotto opzione `nopriv` di `autogroup` non deve essere usata.

La configurazione di `autogroup` è la stessa del modulo `autohome`, ma diversamente da `autohome`, la *directory base virtuale* può essere collocata ovunque e con qualsiasi nome. Non è infatti controllata da account di sistema.

Il modulo `autogroup` può essere usato con **Autodir** con l'opzione `-m`. Per esempio,`-m /usr/lib/autodir/autogroup.so`.

Tutte le sotto opzioni spiegate in gestire le directory home sono valide per `autogroup`, eccetto `skel` e `noskel`,le quali non hanno significato con il modulo `autogroup`. In aggiunta, ci sono delle sotto opzioni specifiche di `autogroup`. Queste opzioni sono spiegate qui sotto.

`nopriv`

Alcune installazioni Linux utilizzano gruppi utente privati Se le directory per questi gruppi non devono essere create, allora occorre utilizzare questa opzione.

20. Le opzioni di autodir

In questa sezione vengono spiegate alcune opzioni di **Autodir**. Le opzioni di backup sono spiegate nella sezione backup.

`-d`

Specifica la *directory base virtuale*. Se il percorso non esiste, verrà creato. Questa opzione si aspetta un percorso assoluto.

`-t`

Intervallo di scadenza per le *directory virtuali*. Per maggiori dettagli riferirsi alla scadenza della directory virtuale.

`-m`

Modulo da usarsi con **Autodir**. Attualmente sono disponibili `autohome` e `autogroup`. Deve essere specificato il percorso completo del modulo.

`-o`

Tutte le sotto opzioni da passare al modulo sono specificate qui. La sintassi da usare per passare le opzioni è simile a quella dell'opzione `-o` del comando **mount**. Vedi le sezioni specifiche dei moduli per maggiori informazioni.

`-f`

Esegue il comando in primo piano e registra tutti i messaggi sulla console. A scopo di debug e per vedere come lavora **Autodir**.

`-l`

Questa opzione specifica il percorso di un file nel quale **Autodir** scriverà il suo ID di processo.

`-h`

La guida a tutte le opzioni supportate da **Autodir**.

`-v`

Informazioni sulla versione di **Autodir**.

21. Opzioni di backup

Queste opzioni sono passate ad **Autodir** per richiedere il supporto al backup.

`-b`

Questa è l'opzione principale per indicare il percorso del programma di backup ed i suoi argomenti. Il percorso deve essere assoluto, altrimenti **Autodir** non lo accetta.

`-w`

Se una *directory virtuale* non è utilizzata per un periodo di tempo, si assume che sia inattiva e viene smontata. Dopo lo smontaggio della directory, questa opzione definisce se il programma di backup viene eseguito immediatamente o dopo un intervallo di tempo. Richiede un argomento in secondi. E' l'intervallo di attesa *minimo* prima di eseguire il backup dopo la scadenza della *directory virtuale*. Non dovrebbe essere più grande di un giorno.

`-p`

E' la priorità da dare al processo di backup. E' compresa tra 1 e 40 inclusi. Un valore più basso indica una priorità più alta e viceversa. Il valore predefinito è 30.

`-c`

Limita il numero di processi di backup attivi allo stesso tempo. Il valore predefinito è 150.

> **Usare gli apici**
>
> L'argomento della opzione `-b` include il percorso assoluto del programma di backup come anche i suoi argomenti. E' quindi consigliato usare i singoli apici per delimitare il suo argomento.

L'opzione `-b` richiede il percorso ad un file eseguibile come anche ad i suoi argomenti. Tuttavia, gli argomenti sono interpreti alla ricerca di sequenze di caratteri %x che vengono sostituiti da stringhe predefinite come segue:

`%N`

Sostituito con il nome della *directory virtuale*.

`%L`

Sostituito con il percorso completo alla *directory reale*.

`%K`

Sostituito con il nome dell'host.

Altre

Altre sequenze sono passate a `strftime`. Vedi le pagine di manuale di `strftime` per maggiori informazioni.

22. Esempi

```
# autodir -d /home       \
  -m /usr/lib/autodir/autohome.so   \
  -t 1000      \
  -f       \
  -o 'realpath=/autohome,level=1,skel=/etc/skel'  \
  -l /var/run/autodir.pid

# autodir -d /home      \
  -m /usr/lib/autodir/autohome.so   \
  -t 300      \
  -b '/bin/tar cf /tmp/%N%F.tar %L'  \
  -w 600      \
  -o 'realpath=/tmp/autohome,level=2,noskel' \
  -l /var/run/autodir.pid

# autodir -d /var/abase/      \
  -m /usr/lib/autodir/autogroup.so  \
  -t 300      \
  -b '/bin/tar cf /tmp/%N%F.tar %L'  \
  -w 86400     \
  -o 'nopriv,nosetgid,realpath=/var/realbase,level=0'
```

23. Particolarità di RPM

Autodir può essere installato da rpm come segue:

```
# rpm -ivh autodir-0.28-4.i386.rpm
```

Se installato da rpm, sono forniti due script di avvio: `/etc/rc.d/init.d/autohome` e `/etc/rc.d/init.d/autogroup`. Il primo serve per eseguire **Autodir** con il modulo `autohome`, il secondo per eseguirlo con il modulo `autogroup`.

Gli script di configurazione `/etc/sysconfig/autohome` e `/etc/sysconfig/autogroup` possono essere usati per specificare quali opzione devono essere passate ad **Autodir**.

24. Ulteriori informazioni

- **Mailing list su autodir** http:// lists.sourceforge.net/mailman/listinfo/intraperson-autodir (http://lists.sourceforge.net/mailman/listinfo/intraperson-autodir).
- Il Sito web ufficiale è http://www.intraperson.com/autodir/ (http://www.intraperson.com/autodir/).
- Autofs mailing list http://linux.kernel.org/ mailman/listinfo/autofs (http://linux.kernel.org/mailman/listinfo/autofs).

- L'HowTo su Automount può essere trovato su http://www.tldp.org
- Autofs Hacking http://www.goop.org/~jeremy/autofs (http://www.goop.org/~jeremy/autofs/).