Alma Mater Studiorum · Università di Bologna

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea in Scienze di Internet

Sincronizzazione e condivisione di file system tra sistemi operativi

Tesi di Laurea in Sistemi Operativi

Relatore:
Chiar.mo Prof.
Davide Sangiorgi

Presentata da:
Marco Gasperini

Sessione II
Anno Accademico 2009-2010

Introduzione

La necessità di sincronizzare i propri dati si presenta in una moltitudine di situazioni, infatti il numero di dispositivi informatici a nostra disposizione è in continua crescita e, all' aumentare del loro numero, cresce l' esigenza di mantenere aggiornate le multiple copie dei dati in essi memorizzati.
Vi sono diversi fattori che complicano tale situazione, tra questi la varietà sempre maggiore dei sistemi operativi utilizzati nei diversi dispositivi, si parla di Microsoft Windows, delle tante distribuzioni Linux, di Mac OS X, di Solaris o di altri sistemi operativi UNIX, senza contare i sistemi operativi più orientati al settore mobile come Android.
Ogni sistema operativo ha inoltre un modo particolare di gestire i dati, si pensi alla differente gestione dei permessi dei file o alla sensibilità alle maiuscole.
Bisogna anche considerare che se gli aggiornamenti dei dati avvenissero soltanto su di uno di questi dispositivi sarebbe richiesta una semplice copia dei dati aggiornati sugli altri dispositivi, ma che non è sempre possibile utilizzare tale approccio.
Infatti i dati vengono spesso aggiornati in maniera indipendente in più di un dispositivo, magari nello stesso momento, è pertanto necessario che le applicazioni che si occupano di sincronizzare tali dati riconoscano le situazioni di conflitto, nelle quali gli stessi dati sono stati aggiornati in più di una copia ed in maniera differente, e permettano di risolverle, uniformando lo stato delle repliche.
Considerando l' importanza e il valore che possono avere i dati, sia a livello lavorativo che personale, è necessario che tali applicazioni possano garantirne la sicurezza, evitando in ogni caso un loro danneggiamento, perchè sempre più spesso il valore di un dispositivo dipende più dai dati in esso contenuti che dal costo dello hardware.
In questa tesi verranno illustrate alcune idee alternative su come possa aver luogo la condivisione e la sincronizzazione di dati tra sistemi operativi diversi, sia nel caso in cui siano installati nello stesso dispositivo che tra dispositivi differenti.
La prima parte della tesi descriverà nel dettaglio l' applicativo Unison.
Tale applicazione, consente di mantenere sincronizzate tra di loro repliche dei dati, memorizzate in diversi dispositivi che possono anche eseguire sistemi operativi differenti.
Unison funziona a livello utente, analizzando separatamente lo stato delle repliche al momento dell' esecuzione, senza cioè mantenere traccia delle operazioni che sono state effettuate sui dati per modificarli dal loro stato precedente a quello attuale.

Unison permette la sincronizzazione anche quando i dati siano stati modificati in maniera indipendente su più di un dispositivo, occupandosi di risolvere gli eventuali conflitti che possono verificarsi rispettando la volontà dell' utente.
Verranno messe in evidenza le strategie utilizzate dai suoi ideatori per garantire la sicurezza dei dati ad esso affidati e come queste abbiano effetto nelle più diverse condizioni.
Verrà poi fornita un' analisi dettagiata di come possa essere utilizzata l' applicazione, fornendo una descrizione accurata delle funzionalità e vari esempi per renderne più chiaro il funzionamento.
Nella seconda parte della tesi si descriverà invece come condividere file system tra sistemi operativi diversi all' interno della stessa macchina, si tratta di un approccio diametralmente opposto al precedente, in cui al posto di avere una singola copia dei dati, si manteneva una replica per ogni dispositivo coinvolto.
Concentrando l' attenzione sui sistemi operativi Linux e Microsoft Windows verranno descritti approfonditamente gli strumenti utilizzati e illustrate le caratteristiche tecniche sottostanti.

Indice

Elenco delle figure

Capitolo 1

Unison

In questo capitolo verranno trattate le procedure per l' installazione, la configurazione e le prove di funzionamento relative all' applicativo Unison.
Unison [1] secondo la definizione dei suoi autori è "un' applicazione per la sincronizzazione di file funzionante sia su Unix e Microsoft Windows che permette di mantenere due repliche di una collezione di file e directory su host (dispositivi) differenti (o su dischi differenti sullo stesso host), modificarli separatamente e sincronizzarli propagando le modifiche apportate in ogni replica nell'altra." [2].
Questa applicazione consente quindi sia di effettuare backup dei dati, nel caso in cui le modifiche siano effettuate su una sola replica, che di effettuare sincronizzazioni, a coppie, tra più repliche nel caso siano state apportate modifiche a più di una replica dall' ultima sincronizzazione.
In quest' ultimo caso Unison richiederà all' utente di specificare come debbano essere risolti gli eventuali conflitti.
È evidente che quando la sincronizzazione avviene tra sistemi operativi diversi possano nascere una serie di problematiche dovute alla diversità del modo in cui vengono memorizzati i dati, basti pensare ai diversi caratteri consentiti nei nomi dei file e delle directory su Linux e su Microsoft Windows, oppure alla sensibilità o insensibilità alle maiuscole, che può causare problemi in presenza di file il cui nome differisca solamente nell' essere scritto in maiuscolo o in minuscolo, situazione ammessa da Linux ma non da Microsoft Windows.
Unison deve essere in grado di gestire situazioni di questo tipo senza incorrere in errori e nel seguito verranno descritte le strategie utilizzate dagli autori per tali casi.

1.1 Il funzionamento di Unison

Nell' articolo "How to build a file synchronizer" [3] vengono illustrate in maniera dettagliata tutte le scelte compiute dagli autori per arrivare allo stato attuale di sviluppo, ponendo l' attenzione su quanto possano essere sottili i problemi in un' applicazione di questo tipo, molte delle informazioni presenti in questa sezione sono tratte da tale articolo.
Gli autori raccontano di come si siano resi conto che sviluppare un file synchronizer richiedeva la definizione di una specifica formale dell' applicazione che ne illustrasse il funzionamento nei minimi dettagli permettendo inoltre di dimostrarne il corretto funzionamento in tutte le situazioni di utilizzo.
L' implementazione finale differisce leggermente dalla specifica formale a causa delle caratteristiche e delle differenze dei diversi sistemi in cui l' applicazione deve poter essere eseguita, ma la formulazione formale ha comunque portato a diversi vantaggi durante lo sviluppo.
La definizione formale e la relativa implementazione in OCaml[1] sono trattate nell' articolo "What's in Unison? A Formal Specification and Reference Implementation of a File Synchronizer" [4], in cui vengono inoltre discussi i vantaggi che tale approccio ha portato nello sviluppo, sia per aver permesso l' identificazione di problematiche che altrimenti sarebbero passate inosservate che per aver garantito una visione comune del funzionamento dell' applicazione tra gli sviluppatori, fondamentale durante l' implementazione.
Questa metodologia di sviluppo ha permesso di migliorare l' affidabilità dell' applicazione, obiettivo fin da subito ritenuto fondamentale per un' applicazione di questo tipo.
Se un file synchronizer non fosse affidabile potrebbe portare a perdite di dati, le quali considerando l' importanza dei dati che possono essere affidati all' applicazione possono risultare particolarmente dannose per gli utenti.
Una importante caratteristica di Unison è l' essere un' applicazione che viene eseguita a livello utente, questo ne permette una maggiore portabilità, come testimoniato dalla vasta gamma di sistemi operativi su cui può essere eseguito.
Questa sua flessibilità ha però lo svantaggio di comportare una minore quantità di informazioni di cui può disporre per identificare in che modo sia cambiato lo stato di una replica nel tempo.
Questo proprio a causa della esecuzione a livello utente e alla conseguente scarsa integrazione con uno specifico sistema operativo o file system.
Per questo motivo Unison si basa solo sullo stato corrente della replica[2] e per tale motivo viene definito un synchronizer basato sullo stato ("state-based").
L' altra tipologia di synchronizer è detta basata sulla traccia ("trace-based"), in quanto può esaminare una traccia completa di tutte le modifiche effettuate all' interno di una

[1] Il linguaggio di programmazione utilizzato per sviluppare Unison.

[2] Nel caso sia disponibile, Unison si basa anche sullo stato relativo all' ultima sincronizzazione effettuata correttamente, tali informazioni vengono memorizzate nei file di archivio.

replica, questo possibilità sarebbe stata indubbiamente un vantaggio per l' applicazione, ma avrebbe richiesto interventi ad un livello più basso che ne avrebbero ridotto la portabilità.
Unison si basa su di un' architettura client-server e può funzionare in due modalità:

- modalità "socket"
- modalità "tunneling".

La scelta tra le due alternative ha importanti risvolti sulla sicurezza, infatti utilizzando la modalità socket la comunicazione tra gli host avviene senza alcuna forma di crittografia, risultando pertanto poco sicura.
L' utilizzo della modalità tunnelling permette invece di utilizzare oltre a RSH, che soffre anch' esso dello stesso problema, anche SSH che facendo uso della crittografia, permette uno scambio di informazioni sicuro tra gli host.
La scelta di una delle due modalità cambia anche la modalità di esecuzione di Unison negli host.
La modalità socket richiede che Unison venga avviato manualmente su entrambi.
Il server viene avviato eseguendo il comando:

$ unison -socket numeroPorta

e il client con il comando:

$ unison root1 socket://nomeHost:numeroPorta/root2

Con root si intende la directory di partenza per la sincronizzazione su un dato host, tale directory e tutte le directory figlie verranno sincronizzate dall' applicazione Unison. Si noti che il percorso root2 deve essere specificato a partire dalla directory in cui è stato eseguito il comando per avviare Unison sul server.
Con la modalità tunnelling solo il client deve eseguire manualmente Unison, mentre sul server esso verrà avviato in maniera automatica.
Durante una sincronizzazione è possibile identificare tre fasi nel funzionamento di Unison:

1. Rilevamento degli aggiornamenti
2. Riconciliazione
3. Propagazione

Ognuna di queste fasi verrà ora illustrata nel dettaglio.

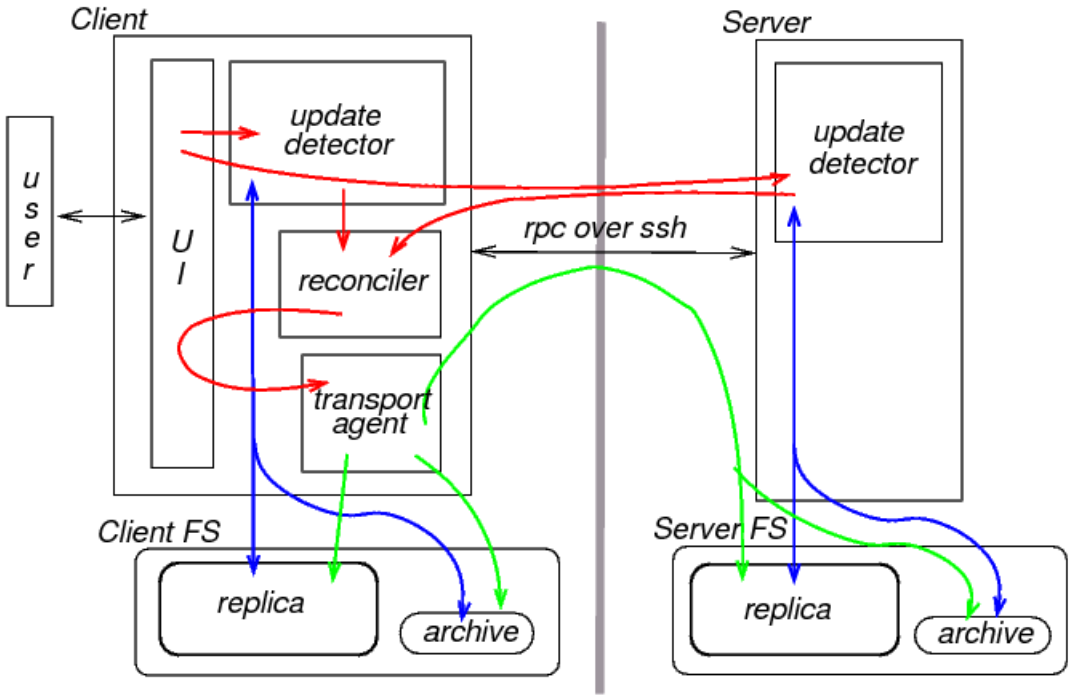

Figura 1.1: Architettura di Unison, la fonte dell' immagine è [3]

1.1.1 Rilevazione degli aggiornamenti

In questa fase viene controllato se vi siano stati aggiornamenti rispetto ad una precedente esecuzione, si controllano pertanto le informazioni contenute in un file archivio che viene mantenuto aggiornato da Unison al termine di ogni sincronizzazione.
Questi file archivio contengono vari metadati e una traccia (fingerprint) del contenuto di ogni file.
Le informazioni presenti nell' archivio vengono confrontate con lo stato attuale della replica.
Nel caso la sincronizzazione sia effettuata in remoto, cioè in presenza di un host che opera come client e di un altro che opera da server, queste informazioni vengono elaborate separatamente e viene creata una lista contenente i percorsi il cui contenuto sia cambiato.
Questa fase termina con l' invio al client della lista di aggiornamenti elaborata dal server. [3]
È necessario definire cosa Unison consideri un aggiornamento, nel manuale [2] vengono fornite le seguenti definizioni.
Per Unison un percorso (path) "è una sequenza di nomi separati dal carattere '/' ".[3]
Tale percorso può riferirsi ad un "file", a una "directory", ad un "collegamento simbolico" o essere "assente", nel caso in cui non si riferisca a nulla all' interno della replica.
Un percorso può essere di tipo assoluto o relativo.
Un percorso assoluto comincia dalla radice del file system, per esempio /dir1/dir2/ in Linux oppure C:\dir1\dir2 in Microsoft Windows.
Un percorso relativo invece comincia da un directory, detta directory corrente e si riconosce perchè non comincia con il carattere '/' su Linux o con la lettera associata ad una unità in Microsoft Windows, per esempio dir1/dir2/ in Linux oppure dir1\dir2 in Microsoft Windows.
Il contenuto di un percorso varia a seconda di ciò a cui si riferisce, nel caso di un file esso è rappresentato dal suo contenuto e dai permessi; se il percorso si riferisce ad un collegamento simbolico il contenuto è rappresentato da una stringa a che specifica che cosa indirizza il collegamento; se si riferisce ad una directory il contenuto è costituito da un token "DIRECTORY" e dai permessi della directory; se infine non si riferisce a nulla il suo contenuto è costituito dal token "ABSENT".
Un' ulteriore definizione che deve essere fornita riguarda l' aggiornamento:
"Si dice che un percorso è aggiornato se il suo contenuto corrente è diverso dal suo contenuto l' ultima volta che è stato sincronizzato con successo."
Dato che Unison controlla il contenuto di un file il semplice cambiamento del tempo dell' ultima modifica, ottenibile per esempio tramite l' utilizzo del comando "touch", non viene riconosciuto come un aggiornamento.

[3]Unison considera solamente '/' come carattere separatore all' interno di un percorso, a prescindere delle convenzioni del sistema operativo in cui viene eseguito.

Nell' identificare un aggiornamento vengono usate strategie diverse a seconda del sistema operativo in uso, ad esempio nel caso dell' implementazione per Unix[4] vengono analizzati il numero dell' inode del file e il suo tempo di modifica, si veda [2] nella sezione "Caveats and Shortcomings".

1.1.2 Riconciliazione

In questa fase vengono fuse le informazioni contenute nelle due liste di aggiornamenti.
Viene ottenuta un' unica lista (detta task list) che contiene le informazioni su quali percorsi siano stati aggiornati segnalando per ognuno di essi in quale direzione debbano essere propagati gli aggiornamenti per rendere nuovamente uguali le repliche.
È qui che particolare attenzione deve essere posta nella gestione dei conflitti.
Si dice, si veda [3] pag 4, che un percorso è in conflitto se:

1. è stato aggiornato in una replica
2. se lui o almeno uno dei suoi discendenti[5] è stato aggiornato nell' altra replica
3. se il suo contenuto nelle due repliche è differente

I percorsi che vengono riconosciuti come in conflitto vengono aggiunti alla task list ma non viene loro attribuita una direzione privilegiata per la propagazione degli aggiornamenti.
In caso di assenza dei file di archivio, situazione che può verificarsi sia nella prima sincronizzazione che viene effettua tra due file system, sia per una cancellazione volontaria da parte dell' utente, Unison definisce come da propagare i file che esistono solo in una replica, in conflitto quelli che esistono in entrambe le repliche e che presentano un contenuto diverso, e come sincronizzati i file presenti in entrambe le repliche con contenuto uguale.
A questo punto viene presentata all' utente, tramite l' interfaccia testuale o grafica, la possibilità di modificare il verso di propagazione degli aggiornamenti, risolvendo così anche eventuali conflitti, e di ignorare l' aggiornamento di certi percorsi.
È possibile anche eseguire Unison con l' opzione "-batch", in tal caso non viene richiesta alcuna interazione da parte dell' utente, attivando tale opzione tutti gli aggiornamenti non in conflitto sono propagati mentre quelli che confliggono sono ignorati.
Si rimanda alla sezione 1.4 per maggiori dettagli sulle opzioni di Unison.

1.1.3 Propagazione

Durante questa fase vengono propagati gli aggiornamenti tra le due repliche, pertanto è in questa fase che devono essere presi tutta una serie di accorgimenti per evitare che i

[4] Quindi ad esempio Linux, Mac OS X, BSD e Solaris.
[5] Cioè dei percorsi che hanno origine da esso.

due file system siano lasciati in uno stato inconsistente.
Vi sono due invarianti che Unison si preoccupa di rispettare per garantire la correttezza del contenuto dei file system di cui si sta effettuando la sincronizzazione.
Tali invarianti risultano fondamentali se si considera che le applicazioni possono incorrere in situazioni di errore come crash, problemi dovuti alla concorrenza, interruzioni da parte dell' utente, rotture dello hardware o errori di comunicazione nella rete e che Unison non deve in alcun caso causare perdite di dati.
Le due invarianti, illustrate in [2] e [3] sono:

1. In ogni momento, ogni percorso in ogni replica può avere o (1) il suo contenuto originario, quindi nessuna modifica vi è stata apportata, oppure (2) il suo contenuto finale, cioè il contenuto che l' utente si aspettava fosse propagato dall' altra replica.

2. In ogni momento, le informazioni memorizzate sul disco riguardo allo stato privato di Unison possono essere o (1) immodificate, o (2) aggiornate per indicare quali percorsi siano stati sincronizzati con successo.

La prima invariante servirebbe a garantire che ogni sincronizzazione interrotta sia una sincronizzazione parziale ma corretta, che può venire completata alla successiva esecuzione di Unison senza errori.
In pratica non è sempre possibile effettuare la sostituzione di file in maniera atomica, cioè garantire che si abbiano sempre o il file originario o il file finale, escludendo quindi situazioni in cui il file originario venga cancellato e non sostituito con il file finale.
Questa impossibilità deriva dalle caratteristiche dei sistemi in cui Unison viene eseguito. Essendo una applicazione eseguita a livello utente la strategia seguita dagli autori è di simulare la sostituzione atomica dei file nei sistemi in cui non sia disponibile utilizzando un commit in due fasi (two-phase commit). [3]
Purtroppo in caso di interruzione in questa fase non è garantito che le invarianti sopra descritte siano rispettate, ma verrebbero ripristinate solamente alla successiva esecuzione di Unison.
La fase termina con l' aggiornamento dei file archivio memorizzati sul client e sul server, in cui vengono aggiunte le informazioni relative ai percorsi che sono stati sincronizzati con successo.

1.2 Configurazione iniziale

In questa sezione si provvederà ad analizzare l' installazione e la configurazione dell' applicativo Unison versione 2.40.xx nei sistemi operativi Microsoft Windows, Mac OS X e Linux.
La versione 2.40.xx risulta in versione beta, ma è stata preferita alla versione stabile per la possibilità di verificare le caratteristiche più aggiornate dell' applicazione, evitando la

maggiore instabilità della versione di sviluppo, disponibile in un repository subversion.
Si intende descrivere il procedimento necessario all' installazione di Unison nei sistemi operativi sopra elencati, illustrando i problemi incontrati e le soluzioni messe in pratica per risolverli.
La fonte principale del seguito del capitolo è lo "User Manual and Reference Guide" relativo alla versione beta di Unison [2].

1.2.1 Microsoft Windows

Si procede ad installare la versione 2.40.16 di Unison compilata da Alan Schmitt [5].
Come segnalato sul sito è necessario installare i pacchetti

- gtk-runtime-2.14.7.0 [6]
- Microsoft Visual C++ 2008 Redistributable Package (x86) [7]

Si deve procedere quindi a decomprimere il file compresso contenente gli eseguibili di Unison [8] in una cartella di propria scelta e inserire nella variabile d' ambiente PATH di Microsoft Windows il percorso completo fino agli eseguibili sia delle librerie grafiche gtk (es. C:\Programmi\gtk-runtime-2.14.7.0\Gtk\bin) che di Unison (es. C:\Programmi\unison-2.40.16-beta).
È inoltre consigliabile rinominare l'eseguibile con interfaccia utente testuale di unison in "unison.exe" per permetterne l' esecuzione in automatico in seguito a un tentativo di sincronizzazione quando si utilizzi la modalità tunnelling.
Si deve poi procedere all' installazione dell' applicazione openSSH, che verrà installata tramite Cygwin.
Cygwin [9] è un' applicazione che utilizzando un layer che emula le API linux e una collezione di strumenti originari di sistemi operativi Unix, permette di compilare ed eseguire applicazioni originariamente sviluppate per Linux su Windows.
L' utilizzo di Cygwin permette l' installazione di OpenSSH [10] che è una collezione di strumenti open source per la connettività basati sul protocollo SSH.
Il client SSH sarà utilizzato per permettere di stabilire connessioni sicure durante funzionamento dell' applicazione Unison mediante l' utilizzo della crittografia.
La possibilità di installare OpenSSH è fornita nella procedura di installazione di Cygwin, verrà infatti offerta la possibilità di selezionarlo tra gli applicativi da installare.
Prima di iniziare la configurazione di openSSH, è buona norma controllare che ogni account utente di Microsoft Windows abbia una password sicura associata, quando si installerà il servizio sshd infatti sarà possibile eseguire un' autenticazione sulla macchina da remoto, la presenza di password insicure costituirebbe un rischio per la sicurezza.
Si esegua quindi nella shell di cygwin il comando "ssh-host-config", assicurandosi di rispondere come segue alle richieste che verranno presentate:

"privilege separation" yes [6]
"create local user sshd" yes
"install sshd as a service" yes
"CYGWIN=" ntsec tty[7]

Si può ora avviare il servizio sshd con il comando:[8]

net start sshd

È quindi necessario importare le informazioni degli utenti e dei gruppi di Microsoft Windows nei file /etc/passwd e /etc/group, in modo che gli sia consentito di autenticarsi tramite ssh:

mkpasswd -l > /etc/passwd
mkgroup -l > /etc/group

Una descrizione accurata del funzionamento di questi comandi è fornita nella "Cygwin User's Guide" [12].
Nel caso in cui sia attivo un firewall è necessario effettuare il port forwarding per la porta su cui verrà contattato il server SSH, per default tale porta risulta essere la 22.[9]
È inoltre necessario inserire nella variabile d' ambiente PATH anche la directory bin di Cygwin (es. C:\Programmi\Cygwin\bin), in tal modo sarà possibile eseguire anche in una shell MS-DOS il comando ssh.
Si può controllare che tutto funzioni correttamente effettuando l' autenticazione allo Host con il comando:

ssh utente@nomeHost

oppure

ssh nomeHost -l utente

[6]La separazione dei privilegi è una misura di sicurezza, viene utilizzata per prevenire la privilege escalation (letteralmente la scalata dei privilegi), pratica che permette ad un utente di ottenere privilegi superiori sfruttando errori nello sviluppo di un' applicazione.

[7]Con le versioni più recenti di Cygwin è consigliabile utilizzare l' opzione "acl" piuttosto che le opzioni "ntsec tty", per maggiori informazioni si veda la "Cygwin User's Guide". [11]

[8]Da eseguire in una shell MS-DOS o di Cygwin.

[9]Si controlli anche l' applicazione Windows Firewall.

1.2.2 Linux

Non essendo disponibile alcun pacchetto binario per la versione 2.40.16 si è provveduto a compilarlo.
La procedura più generale consiste nello scaricare il sorgente della versione desiderata dal sito di unison.[10] [13]
Se si utilizzano Debian o Ubuntu si può procedere ad installare tutte le dipendenze richieste con:

sudo apt-get build-dep unison[11]

Altrimenti si può utilizzare il metodo standard suggerito dal manuale [2] per installare le dipendenze richieste.

Si decomprime il pacchetto coi sorgenti:

tar -xvf unison-2.40.16.tar.gz
cd unison-2.40.16/

Si procede alla compilazione dei sorgenti:

make

È possibile forzare la tipologia di interfaccia grafica desiderata passando a make il parametro UISTYLE=text/gtk/gtk2.[12]
Per poter raggiungere l' eseguibile di Unison semplicemente scrivendo il comando "unison" e per permetterne l' esecuzione automatica quando l' host su cui è stato installato agisca come server, si è creato un collegamento simbolico nella cartella /usr/bin/ all' eseguibile di unison con interfaccia testuale con il comando:

```
# ln -s /percorso/unison-2.40.16/unison /usr/bin/unison
```

Per potere utilizzare un host come server per Unison è inoltre richiesta l' installazione del pacchetto "openssh-server", come descritto nella sezione 1.1 infatti l' utilizzo di Unison su un client in modalità tunneling fa si che venga avviato un processo di Unison

[10] Nel seguito si utilizzerà il file unison-2.40.16.tar.gz
[11] Si è provveduto anche ad installare il pacchetto dh-ocaml versione 0.9.3.
[12] Per esempio, per compilare la versione con interfaccia grafica va eseguito:
make UISTYLE=gtk2
Si noti che durante la compilazione viene automaticamente impostata l' opzione UISTYLE=gtk2 se possibile.

sul server tramite ssh.
L' installazione di openssh-server non è invece richiesta se l' host si vuole utilizzare solamente come client, solitamente le distribuzioni Linux forniscono il client di ssh preinstallato.

1.2.3 Mac OS X

Si è provveduto ad installare la versione 2.40.1 di Unison compilata da Alan Schmitt [5] che contiene tutto il necessario per l' esecuzione dell' applicazione.
Al primo avvio verrà chiesto se installare la versione testuale di Unison, si consiglia di effettuarne l' installazione, che è comunque accessibile in qualunque momento da "Unison → Install Command line tool".
Per utilizzare come server per Unison su Mac OS X è necessario abilitare il login remoto da:
"Preferenze di Sistema" → "Condivisione" e in "Internet e Wireless" abilitare "Login Remoto".

1.3 Prove di funzionamento

In questa sezione verranno illustrate le prove di funzionamento per effettuare la sincronizzazione di file system tra i sistemi operativi Windows, Linux, Mac OS X utilizzando l' applicazione Unison.
Verranno illustrate le configurazioni necessarie per consentire una sincronizzazione senza errori tra i sistemi operativi in esame, sottolineando quali delle loro caratteristiche potrebbero causare problemi.
Le prove verranno effettuate utilizzando le configurazioni di Unison presentate nella sezione 1.2, utilizzando sia l' interfaccia grafica che quella testuale.
A seconda dell' eseguibile utilizzato può cambiare l' interfaccia grafica predefinita, si può forzare l' utilizzo dell' alternativa eseguendo Unison con l' opzione "ui", tale opzione accetta i valori "text" e "graphic", quindi per eseguire Unison con l' interfaccia testuale si può utilizzare il comando:[13]

```
$ unison -ui text
```

Nelle prove descritte in questa sezione si è utilizzata la modalità di connessione tunneling tramite ssh, tale modalità risulta quella consigliata in quanto, come si è già riportato nella sezione 1.1, in questo modo viene utilizzata la crittografia per aumentare la sicurezza durante la trasmissione dei dati.
Si consideri che con la sintassi "SistemaOperativo1 → SistemaOperativo2" si intende che l' host che esegue il SistemaOperativo1 viene utilizzato in modalità client, attraverso la connessione ssh Unison avvierà automaticamente un processo server sull' host che esegue il SistemaOperativo2 senza bisogno di interventi da parte dell' utente.
Il risultato delle prove è che Unison risulta funzionare correttamente su tutti i sistemi operativi menzionati, difatti quando viene rilevata una possibile fonte di errore, come una differenza di sensibilità rispetto alle maiuscole tra i sistemi operativi coinvolti, Unison esclude dalla sincronizzazione i file che potrebbero causare problemi.

[13] Solitamente un eseguibile che fornisce l' interfaccia grafica permette anche l' utilizzo di quella testuale, mentre non è sempre vero il contrario.

1.3.1 Linux → Mac OS X , Linux → Windows

L' eseguibile di Unison con cui sono state effettuate le prove con il sistema operativo Linux utilizza come predefinita l' interfaccia testuale, pertanto pertanto per utilizzare da tale interfaccia è sufficiente eseguire il comando unison.
Per effettuare la sincronizzazione si può utilizzare il comando:

unison percorso/da/sincronizzare/ ssh://nomeUtente@nomeHost/percorso/da/sincronizzare

dove nomeHost può essere sia il nome assegnato ad un host che il suo indirizzo IP.

Per esempio, utilizzando Mac OS X come server potrebbe essere:

unison Scrivania/unison1/ ssh://nomeUtente@indirizzoIP/Desktop/unison2

oppure utilizzando Windows come server:

unison Scrivania/unison1/ ssh://nomeUtente@indirizzoIP/C:/Documents\
and\ Settings/nomeUtente/Desktop/unison2[14]

È anche possibile effettuare la sincronizzazione utilizzando l' interfaccia grafica.
È necessario creare un profilo che in seguito potrà essere utilizzato per utilizzare uno specifico insieme di opzioni di Unison senza doverle inserire ad ogni sincronizzazione.
Si deve quindi creare un nuovo profilo inserendo il nome desiderato, successivamente si sceglie la modalità di sincronizzazione desiderata, per quanto esposto nella sezione 1.1 si consiglia di selezionare "using SSH".
Il passo successivo consiste nell' inserire il nome dell' utente e dell' host remoto.
Devono poi essere inserite anche le directory root di cui sarà effettuata la sincronizzazione.[15]
Infine va specificato se attivare il supporto specifico per partizioni FAT, che in questo caso non risulta necessario.
Una volta che il profilo è stato creato è sufficiente selezionarlo dalla interfaccia grafica e scegliere "apri" per effettuare la sincronizzazione.
Si noti che i parametri "First Root" e "Second Root" che appaiono selezionando il profilo sono gli stessi parametri che andrebbero passati a Unison se avviato con l'interfaccia testuale.

[14] Il comando deve essere scritto su un' unica riga, viene riportato in questo modo solo per motivi di leggibilità.
[15] Prestando attenzione a non specificare il percorso della directory remota tra i caratteri ""

1.3.2 Mac OS X → Linux , Mac OS X → Windows

Il pacchetto binario utilizzato fornisce come impostazione predefinita l' interfaccia grafica, per utilizzare l' interfaccia testuale risulta pertanto necessario il passaggio dell' opzione "-ui text" all' eseguibile di unison.
Il comando da impartire risulta quindi essere:

unison -ui text percorso/da/sincronizzare/ ssh://nomeUtente@nomeHost/percorso/da/sincronizzare

Per esempio:

unison -ui text Desktop/unison/ ssh://utente@indirizzoIP/Scrivania/unison

L' interfaccia grafica seppur diversa nell' aspetto è equivalente a quella descritta nella sezione 1.3.1, si rimanda pertanto a tale sottosezione per i dettagli.

1.3.3 Windows → Linux , Windows → Mac OS X

Il pacchetto binario installato contiene due eseguibili, uno con interfaccia grafica e l' altro con interfaccia testuale.
Non sono necessarie configurazioni particolari, si rimanda pertanto a quanto descritto nella sezione 1.3.1.

1.3.4 Linux , Mac OS X → partizione FAT accessibile da un sistema operativo diverso da Microsoft Windows

Se i dati da sincronizzare sono memorizzati in un file system FAT[16] accessibile da un sistema operativo diverso da Microsoft Windows,[17] è richiesto l' utilizzo dell' opzione "-fat" o la specifica manuale delle opzioni che contiene.
Tale opzione raccoglie al suo interno tutte le configurazioni necessarie a prevenire situazioni di errore su tale tipologia di file system, ed è equivalente all' utilizzo delle seguenti opzioni:

-perms=0 -dontchmod=true -ignorecase=true -links=false -ignoreinodenumbers=true

[16] File Allocation Table

[17] I file system di tipo FAT sono accessibili su una grande varietà di sistemi operativi, nel capitolo 2 verrà descritto come l' accesso ai dati in essi memorizzati può essere gestito in maniera avanzata dal sistema operativo Linux.

Il significato di queste opzioni è:

- "perms=0" impedisce a Unison di sincronizzare i permessi di file e directory, tale funzionalità non è infatti supportata dai file system di tipo FAT.
- "dontchmod=true" evita che venga effettuata la chiamata di sistema chmod richiesta per cambiare i permessi dei file.
- "ignorecase=true" i nomi dei file vengono trattati in maniera non sensibile alle maiuscole, di conseguenza file il cui nome differisce solo per la presenza di maiuscole vengono trattati come lo stesso file, solitamente questa opzione assume valore true se uno degli host che vengono sincronizzati utilizza Microsoft Windows o Mac OS X.
- "links=false" impedisce che Unison effettui la sincronizzzione dei collegamenti simbolici.
- "ignoreinodenumbers=true" evita che Unison utilizzi i numeri degli inode come metodo per identificare la presenza di aggiornamenti nelle repliche.

1.4 Funzionamento Avanzato

Nei paragrafi precedenti è stato illustrato il funzionamento dell' applicativo Unison e la sua configurazione per ottenere una installazione funzionante sistemi operativi in esame. Obiettivo di questo paragrafo sarà invece descrivere le funzioni più avanzate ottenibili tramite l' utilizzo di configurazioni di uso meno comune, ma che possono risultare utili in condizioni particolari.
Verrà quindi descritto sia come configurare un comportamento temporaneo dell' applicativo specificando le opzioni all' esecuzione del programma, che come possano essere creati dei profili che permettano invece di rendere permanenti tali opzioni anche nelle successive operazioni di sincronizzazione.
Ulteriore argomento che verrà trattato in questo paragrafo sarà come effettuare nel modo migliore la sincronizzazione tra più di due dispositivi.

1.4.1 Le preferenze di Unison

Unison fornisce una serie di opzioni che permettono di modificarne il comportamento durante la sincronizzazione dei dati, l' ordine con cui sono specificate è irrilevante e possono anche essere intervallate al nome del profilo e/o alla specifica delle directory root.[18]
Sono presenti tre tipi di opzioni, suddivise in base al tipo di valore che accettano:

- Le opzioni che accettano valori booleani possono essere specificate aggiungendo come argomento di Unison "-nomeOpzione", che è equivalente a scrivere "-nomeOpzione=true", o nel caso in cui vi si voglia assegnare valore false "-nomeOpzione=false".
- Le opzioni che accettano valori numerici possono essere specificate aggiungendo "-nomeOpzione n".
- Le opzioni che accettano valori di tipo stringa di caratteri possono essere specificate aggiungendo "-nomeOpzione valore" e possono essere utilizzate più volte nello stesso comando.

Tali opzioni possono essere specificate come argomento all' eseguibile di Unison ed in tal caso la loro durata sarà limitata alla singola esecuzione dell' applicazione, oppure essere aggiunte ad un profilo di Unison per renderle permanenti.

[18] Si noti che un nome di profilo può essere specificato come argomento ad Unison insieme alle due directory root.
Questo è possibile solo quando non sia presente una specifica delle directory root in tale profilo.
In caso contrario si incorrerà in un messaggio d' errore.

1.4.1.1 Modifica manuale dei profili

Un profilo è un file di testo al cui interno sono specificati i parametri necessari per una esecuzione di unison, l' aggiunta di un' opzione ad un profilo permette di automatizzare il comportamento di unison, pertanto basta eseguire:

unison nomeProfilo

per avere tutte le preferenze configurate, cioè senza doverle specificare come argomento ogni volta.
Per aggiungere un' opzione al profilo è necessario scrivere all' interno del file .unison/nomeProfilo.prf[19] una riga che rispetti la sintassi:

nomeOpzione = valore

All' interno di un profilo è possibile scegliere se specificare o meno le directory root a cui dovranno applicarsi tali opzioni, in caso si decida di non inserirle sarà possibile specificarle manualmente da riga di comando, risulta così possibile utilizzare le opzioni contenute nel profilo per diverse directory root.
Viene ora proposta una selezione delle opzioni che si ritiene possano risultare maggiormente utili, di esse verranno fornite una descrizione approfondita del funzionamento ed esempi pratici di utilizzo.

1.4.1.2 Automatizzare la sincronizzazione

In molte situazioni può essere desiderabile ridurre la quantità di domande che vengono rivolte all' utente durante l' esecuzione di Unison.
Una situazione comune potrebbe essere per esempio l' utilizzo di Unison in script eseguiti automaticamente dal sistema, realizzare tali script o fare eseguire in automatico dal sistema determinati comandi è solitamente possibile su ogni sistema operativo.
Unison fornisce tra l' altro l' opzione "repeat valore" che consente di eseguire una nuova sincronizzazione dei dati al passare del numero di secondi specificato in maniera automatica, pertanto è ragionevole supporre che l' utente non sarà sempre a disposizione per esprimere la sua opinione sulle operazioni compiute dall' applicazione ed è quindi necessaria una modalità di esecuzione da esso svincolata.
Tale modalità di esecuzione è ottenibile tramite l' utilizzo dell' opzione "batch" di Unison, tale opzione consente di effettuare in maniera automatica la propagazione degli aggiornamenti che non risultano in conflitto, mentre gli altri vengono omessi.
Unison mantiene aggiornato un file di unison.log, tale file viene memorizzato per impostazione predefinita nella directory home dell' utente, ma è possibile cambiarne la

[19] La directory .unison contiene oltre ai profili anche i file di archivio.

posizione con l' opzione "logfile percorso/del/logFile".
L' opzione "silent" permette di stampare a video solo i messaggi che segnalano situazioni di errore, come i conflitti, e attiva automaticamente l' opzione "batch", che pertanto non è necessario aggiungere manualmente.
Nel caso si desiderino sincronizzare directory contenenti molti file, con le impostazioni predefinite Unison chiede all' utente se desidera sovrascrivere la direzione della propagazione degli aggiornamenti per ogni percorso aggiornato.
In questo caso si può desiderare di continuare a interagire con l' applicazione, riducendo però il numero di domande che vengono poste.
L' utente può utilizzare l' opzione "auto", in questo modo verrà chiesto all' utente solamente di risolvere eventuali situazioni di conflitto, mentre in caso non si verifichino viene confermata automaticamente la direzione predefinita per la propagazione, alla fine verrà comunque richiesto all' utente se desidera che tali aggiornamenti siano effettivamente propagati.

1.4.1.3 Specifica di singoli percorsi

Unison fornisce all' utente la possibilità di specificare alcuni percorsi per cui si desidera un trattamento diverso rispetto alla directory root che li contiene.
Questo permette di escludere dalla sincronizzazione determinati file, directory o collegamenti pur mantenendoli memorizzati in una delle repliche, oppure di rendere più veloce la sincronizzazione limitandola ai soli percorsi specificati.
La specifica di tali percorsi avviene attraverso l' utilizzo di pattern, in pratica vengono selezionati tutti i percorsi che corrispondono ad un determinato pattern.
I pattern possono essere specificati sia da riga di comando che all' interno di un profilo ed essere definiti in diverse forme.
Il manuale [2] descrive accuratamente come può essere definito un pattern:

- Espressione regolare, deve essere preceduto dalla parola chiave "Regex".
- Nome, seleziona tutti i percorsi che terminano con il nome specificato, deve essere preceduto dalla parola chiave "Name"
- Percorso, identifica uno specifico percorso e deve essere preceduto dalla parola chiave "Path"
- Un percorso e tutti i suoi figli sono selezionabili utilizzando la parola chiave "BelowPath"

"Name" e "Path" non accettano espressioni regolari, ma accettano i seguenti caratteri ed espressioni:

- * può essere utilizzato per selezionare una qualunque sequenza di caratteri che non contenga '/'[20] e che non inizi con '.' nel caso sia utilizzato con per un "Name".
- ? può essere utilizzato per selezionare un singolo carattere diverso da '/' o da un '.' iniziale.
- [xyz] può essere utilizzato per individuare un qualunque carattere nell' insieme {x,y,z}
- {a,bb,ccc} può essere utilizzato per individuare "a" , "bb" o "ccc".

Sono diverse le opzioni di Unison che richiedono di specificare uno o più percorsi.
Tra queste vi è l' opzione "path percorsoDaSincronizzare" che permette di sincronizzare solo un percorso, o un insieme di percorsi, se ripetuta più volte all' interno dello stesso comando o profilo, e i relativi figli.
Tale opzione permette quindi di tralasciare i percorsi che non sono ritenuti interessanti dall' utente, con il risultato di velocizzare le operazioni di sincronizzazione, soprattutto se le repliche sono di grandi dimensioni.
L' opzione "path" non accetta come argomento delle espressioni regolari, ma soltanto dei nomi di percorso validi.

1.4.1.4 Ignore

Spesso i diversi sistemi operativi, o le applicazioni utilizzate per modificare i dati, memorizzano nei file system file che non sono rilevanti per l' utente.
Può quindi capitare che non si voglia che tale file vengano propagati verso le altre repliche da Unison nelle operazioni di sincronizzazione.
In questi casi risulta utile l' opzione "ignore percorsoDaIgnorare", questa opzione risulta particolarmente utile se inserita nei profili per tralasciare tutti i file con certi caratteri nel nome, nomi o estensione, ma può anche essere utilizzata da riga di comando.
L' opzione "ignore" accetta i pattern per specificare i percorsi da ignorare, i percorsi che vengono selezionati in quanto corrispondenti ad un pattern vengono ignorati e con essi ogni loro percorso figlio.
Unison fornisce inoltre una modalità automatica per impostare pattern per l' opzione "ignore" nel profilo corrente, quando viene presentata all' utente la possibilità di intervenire manualmente esso può per ogni percorso modificato inserire uno dei seguenti

[20] In quanto tale carattere viene interpretato da Unison come separatore nei percorsi a prescindere dal sistema operativo in cui viene eseguito.

caratteri:

- 'I' viene aggiunto il percorso al profilo corrente in modo che sia ignorato in maniera permanente.
- 'E' permette di ignorare tutti i file con la stessa estensione
- 'N' permette di ignorate tutti i percorsi che terminano con quel nome.

È possibile utilizzare l' opzione backup da riga di comando come segue:

unison root0 root1 -ignore 'pattern'

per esempio, per ignorare il percorso nomePercorsoDaIgnorare si può usare il comando:

unison root0 root1 -ignore 'Name nomePercorsoDaIgnorare'

oppure modificando manualmente un profilo, aggiungendo:

ignore = pattern

per esempio, per ignorare lo stesso percorso:

ignore = Name nomePercorsoDaIgnorare[21]

si possono inoltre utilizzare pattern basati su "Path", "BelowPath" e "Regex".

Per evitare che vengano sincronizzati dei file che risulterebbero inutili se le repliche sono gestite utilizzando sistemi operativi diversi, si possono inserire nel profilo in uso le seguenti preferenze:

ignore = Name *~
ignore = Name Thumbs.db
ignore = Name .DS_Store

*~ sono file utili solamente su Linux, Thumbs.db è un file utile solamente su Microsoft Windows e .DS_Store è invece utile solamente su Mac OS X.
Se si utilizzano applicazioni che salvano dei file automaticamente e non si ritiene utile propagarli alle altre repliche si possono ignorare in modo analogo.

[21] nomePercorsoDaIgnorare può anche contenere spazi, questi non causano problemi.

1.4.1.5 Backup

Unison permette di mantenere backup dei file presenti in una replica quando vengono sovrascritti propagando quelli che sono stati aggiornati in un' altra.
È possibile attivare il backup dei file sia per una singola esecuzione di Unison, da riga di comando, sia modificando un profilo.
Per effettuare i backup di ogni file contenuto nella directory "dir1" si può inserire nel profilo:

backup = Path dir1/*

Mentre da riga di comando sarebbe:

unison root0 root1 -backup 'Path dir1/*'

oppure si può effettuare un backup dei file con un dato nome utilizzando la parola chiave "Name", per esempio per effettuare il backup dei soli file il cui nome inizia per 'b' si può inserire nel profilo:[22]

backup = Name b*

I backup possono essere memorizzati nel percorso del file originale (local) o in un percorso unico (central), come impostazione predefinita viene utilizzata la modalità central, ma si può utilizzare l' opzione "backuplocation" (che accetta i valori "local" e "central") per cambiarla.
La directory in cui vengono memorizzati tutti i backup è come impostazione predefinita .unison/backup nella directory home dell' utente, ma tale impostazione può essere modificata specificando una directory differente con l' opzione "backupdir".
Sono inoltre presenti opzioni per cambiare il numero massimo di versioni di un file che si vuole siano mantenute da Unison ("maxbackups numero") e opzioni per modificare il nome che deve essere assegnato ai backup:

- "backupprefix" permette di impostare quale scritta verrà apposta prima del nome del file
- "backupsuffix" permette di impostare quale scritta seguirà il nome del file

è possibile utilizzarle entrambe insieme, ma se si modifica "backupprefix" almeno una tra le due opzioni deve specificare la versione del backup, cioè $VERSION, altrimenti tutti i backup di un file avrebbero lo stesso nome e questo non è consentito.

[22] Per quanto riguarda la specifica di questa opzione da riga di comando vale quanto affermato in precedenza.

La forma varia leggermente dall' utilizzo da riga di comando o da profilo.
Da riga di comando infatti la variabile $VERSION deve essere specificata '$VERSION' come nell' esempio seguente:

-backupprefix ulteriorePrefisso'$VERSION'

mentre il file di profilo non richiede tale accorgimento:

backupprefix = ulteriorePrefisso$VERSION

Il nome che viene impostato in maniera predefinita è nella forma ".bak.VERSION.FILENAME".

1.4.2 Utilizzare Unison per sincronizzare più di due host

Il manuale di Unison nella sezione "Using Unison to Synchronize More Than Two Machines" [14] sostiene che:

> "Unison è progettato per sincronizzare coppie di repliche.
> È comunque possibile utilizzarlo per mantenere sincronizzati
> gruppi più grandi di dispositivi eseguendo sincronizzazioni
> mirate tra coppie di essi.
> La strategia più affidabile per effettuarne la sincronizzazione
> è eseguire le sincronizzazioni organizzando i dispositivi in una
> topologia a stella.
> Si dovrà designare uno dei dispositivi come "hub" e ognuno
> degli altri verrà sincronizzato solamente con l' hub.
> Il vantaggio principale di una topologia a stella è di eliminare la
> possibilità che si verifichino degli "spurious conflicts",[23]
> che possono accadere in quanto Unison mantiene un archivio
> separato per ogni coppia di host che sincronizza."

[23] Per "spurious conflict" si intende una situazione in cui vengano apportate ulteriori modifiche ad un file prima che un suo aggiornamento sia stato propagato a tutte le repliche.

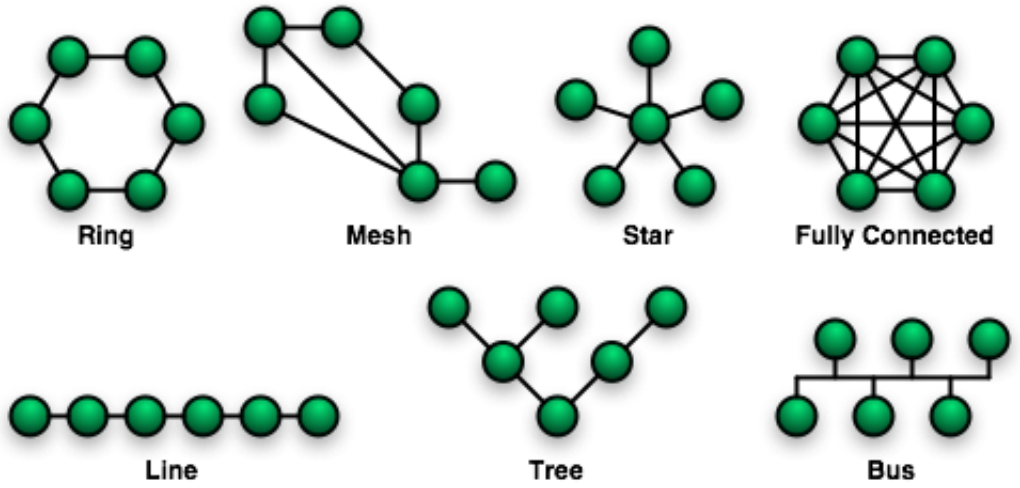

Figura 1.2: Diverse topologie di rete, la fonte dell' immagine è [15]

Capitolo 2

Sincronizzazione di file system tra i sistemi operativi Microsoft Windows e Linux sullo stesso calcolatore

Un approccio alternativo al mantenere sincronizzate repliche degli stessi dati su diversi dispositivi, consiste nel mantenerne memorizzata un' unica copia, a cui avranno accesso diversi dispositivi.
Un caso particolare di tale approccio alternativo, si riscontra nel momento in cui si voglia accedere ai medesimi dati da sistemi operativi diversi installati nella stesso calcolatore.
In questo capitolo verrà descritto come rendere accessibile un' unica copia dei dati a più sistemi operativi.
In particolare si procederà a descrivere come effettuare una condivisione di file system, in cui saranno memorizzati tali dati, tra i sistemi operativi Linux e Microsoft Windows.
Viene ora illustrata la terminologia che verrà utilizzata per il resto del capitolo.

2.1 Terminologia

Avviando una shell testuale in una distribuzione Linux vengono presentate all' utente diverse informazioni, che solitamente corrispondono a nome dell' utente con cui si è effettuato l' accesso, il nome della macchina in cui si è effettuata l' autenticazione dell' utente e la directory corrente.
Per presentare tali informazioni viene solitamente utilizzata una sintassi simile a questa:

"nomeUtente@nomeHost:directoryCorrente"

Solitamente viene inoltre segnalata anche la tipologia di utente con cui è stato effettuato l' accesso.

Se l' autenticazione è stata effettuata da un' utente normale, che presenta quindi privilegi limitati, viene solitamente stampato il carattere '$' al termine della stringa precedente, altrimenti in caso di autenticazione dall' amministratore di sistema viene solitamente stampato il carattere '#'.
Solamente l' utente amministratore, detto anche root o superuser, potrà eseguire qualunque tipo di operazione sul sistema.
Nel seguito del capitolo verranno spesso inseriti dei comandi, a seconda che vi sia anteposto il carattere '$' o '#' si vuole identificare una diversa tipologia di utente che deve eseguire tale comando.
Se viene anteposto il carattere '$' si vuole indicare che tali comandi possono essere eseguiti da un qualunque utente normale, se è invece anteposto il carattere '#' significa che sono necessari i privilegi di amministrazione.
Per eseguire un comando con i privilegi di amministrazione è possibile procedere in diversi modi:

- si può utilizzare il comando "su" e inserire la password dell' account root, in modo l' utente diventa temporaneamente l' utente root.
  Questa soluzione però non funziona su tutte le distribuzioni linux, in alcune l' account di root è disabilitato per motivi di sicurezza, e richiede la conoscenza della password di root, che dovrebbe invece essere diffusa il meno possibile tra gli utenti. Per questi motivi è preferibile utilizzare la soluzione seguente, che presenta proprio il vantaggio del non richiedere che utenti con privilegi limitati conoscano la password dell' account root.

- utilizzare il comando sudo, con cui si possono eseguire comandi come un altro utente, in questo caso root, inserendo la password dell' utente corrente.
  A seconda di come viene configurato, un utente può essere autorizzato ad eseguire ogni comando come se fosse l' altro, oppure ad eseguire solo alcuni comandi specifici.
  L' utilizzo di sudo permette una gestione molto più fine dei permessi, infatti autenticandosi come root un utente può compiere qualunque operazione sul sistema, mentre l' utilizzo di sudo permette di concedere agli utenti l' autorizzazione ad eseguire solo comandi specifici con i privilegi di amministrazione.

2.2 I permessi dei file in Linux

In questa sezione verranno descritti i concetti fondamentali riguardanti l' uso dei permessi in Linux.
Linux distingue i permessi tra tre tipologie di utenti:

- Il proprietario
- Il gruppo
- Tutti gli altri utenti

Per ciascuna di queste categorie di utenti è possibile specificare separatamente i permessi di cui dispongono sui file.[1]
Durante la creazione di un file (per esempio "nomeFile") da parte di un utente esso ne diventa automaticamente il proprietario e il gruppo, essi possono comunque venire modificati utilizzando il comando chown.
Solo l' utente root può cambiare il proprietario del file, mentre il gruppo può essere cambiato anche dal proprietario purchè appartenga sia al gruppo di partenza che a quello di arrivo.
Per esempio si consideri un file appena creato dall' utente "utente1", per quanto appena affermato tale file avrà sia come proprietario che come gruppo l' utente "utente1".
Volendo si possono visualizzare tali informazioni con il comando:

```
$ ls -l nomeFile
```

Il significato delle altre informazioni che vengono riportate da tale comando verranno illustrate più avanti in maniera approfondita, per adesso è sufficiente osservare la terza e la quarta colonna che riportano il proprietario e gruppo di tale file.

Se si volesse cambiare il gruppo sarebbe possibile eseguire[2]:

```
chown :nomeGruppo nomeFile
```

mentre per cambiare il proprietario è richiesto essere autenticati come root:

```
# chown utente1 nomeFile
```

[1] Si noti che in Linux anche le directory sono file.
[2] Autenticati come utente1 o come root

si possono anche effettuare entrambe le operazioni in un unico comando:

chown utente1:nomeGruppo nomeFile

I permessi più comunemente utilizzati sono quelli di Lettura (Read), Scrittura (Write) ed Esecuzione (Execute), il loro significato cambia a seconda che siano applicati a file o a directory.
Se applicati ad un file i permessi hanno il seguente significato: [15] [16]

- Il permesso di Lettura permette di leggere il contenuto del file.
- Il permesso di Scrittura permette di cambiare il contenuto del file.
- Il permesso di Esecuzione permette di eseguire un file eseguibile o uno script.

Se sono invece applicati ad una directory il significato è il seguente:

- Il permesso Lettura permette di elencare il contenuto della directory.
- Il permesso Scrittura permette di aggiungere o cancellare file dalla directory.
- Il permesso Esecuzione permette di attraversare la directory, cioè utilizzare il comando "cd".

A questi si aggiungono altri tre permessi:

- Il permesso Sticky che può essere applicato alle directory e permette la cancellazione o la rinominazione di un file in essa contenuto solo al proprietario del file, al proprietario della directory a cui è applicato il permesso Sticky o all' utente root. Questa limitazione risulta utile quando si vuole impedire agli utenti che possiedono il permesso di scrittura sulla directory di cancellare file di cui non sono proprietari.
- Il permesso Setuid che permette di eseguire un file eseguibile su cui è stato applicato con i permessi del suo proprietario.
- Il permesso Setgid che se viene applicato a un file eseguibile ne permette l' esecuzione con i permessi del gruppo, mentre se è applicato ad una directory fa si che ogni nuovo file che vi viene creato appartenga al medesimo gruppo che possiede la directory.

Viene ora illustrato come si possono elencare i permessi dei file, in seguito verranno definite le due notazioni che possono essere utilizzate per modificarli, la simbolica e la ottale.

Per visualizzare gli attributi dei file[3] si può utilizzare il comando:

$ ls -l nomeFile[4]

si ottiene un output nella forma:

trwxrwxrwx n proprietario gruppo dimensione data file

t indica il tipo di file, nel sistema operativo Linux ve ne sono 7 tipi differenti.
La tabella seguente è tratta da [16] pag. 133

Indicatore di tipo	Tipo di file
-	Regular File
d	Directory
l	Link
c	Character Device
s	Socket
p	Named pipe
b	Block device

I più rilevanti per gli obiettivi di questa tesi sono:
Regular File, che sono file di testo, file di dati o eseguibili binari.
Directory che sono elenchi di file.
Link che si distinguono in hard link e soft link.

Informazioni aggiuntive possono essere reperite con il comando "info ls" alla descrizione dell' argomento "-l".

Seguono tre terne di "rwx", la prima è relativa al "proprietario" del file (owner[5]), la seconda al "gruppo" (group) e la terza a tutti gli "altri" utenti che abbiano accesso al file (others).
Oltre ai valori "rwx" per ciascuna cifra può essere assunto il valore "-" ad indicare che tale permesso non è presente.
Se è impostato il permesso Sticky esso verrà segnalato con il carattere 't' al posto della

[3]Tra i quali vengono elencati anche i permessi.

[4]Per le directory conviene aggiungere anche l' opzione "-d", che corrisponde a utilizzare il comando:
ls -ld nomeDirectory
infatti senza questa opzione verrebbero presentate le informazioni relative al contenuto della directory e non alla directory stessa.

[5]Denominato user in notazione simbolica.

'x' della terna relativa agli "altri" utenti se è attivo anche il permesso di esecuzione per gli "altri", in caso contrario viene riportato il carattere 'T'.
Se è impostato il permesso Setuid esso verrà segnalato con il carattere 's' al posto della 'x' della terna relativa al "proprietario" se è attivo anche il permesso di esecuzione per il proprietario, in caso contrario viene riportato il carattere 'S'.
Lo stesso discorso è applicabile al permesso Setgid, in cui i caratteri 's' o 'S' vengono riportati secondo lo stesso criterio nella terna relativa al gruppo.

Viene ora illustrato come sia possibile operare modifiche ai permessi dei file utilizzando il comando chmod nella notazione numerica e simbolica.
La notazione numerica richiede l' inserimento di tre cifre ottali, la prima corrisponderà ai permessi per il proprietario, la seconda per il gruppo e la terza per gli altri.
In maniera opzionale è possibile farle precedere da una quarta cifra che permette di impostare i permessi Sticky, Setuid, Setgid.
La tabella seguente specifica il valore dei permessi:

Permesso	Valore
Lettura	4
Scrittura	2
Esecuzione	1

Per determinare il valore numerico corrispondente ai permessi desiderati per il proprietario, il gruppo o gli altri è sufficiente sommarne il valore.
Si possono quindi inserire i seguenti valori:

r+w+x = 4+2+1 = 7
r+w = 4+2 = 6
r+x = 4+1 = 5
r = 4
w+x = 2+1 = 3
w = 2
x = 1
- = 0

Per quanto riguarda la quarta cifra opzionale, il suo valore può essere determinato sommando quelli presentati nella tabella seguente:

Permesso	Valore
Setuid	4
Setgid	2
Sticky	1

Per esempio se si volessero concedere i permessi di lettura, scrittura ed esecuzione al proprietario di un file permettendone lettura al gruppo e agli altri, si potrebbe utilizzare:

$ chown 744 nomeFile

mentre per applicare il permesso Setuid concedendo permessi di lettura, scrittura ed esecuzione a tutti gli utenti, si sarebbe dovuto utilizzare:

$ chown 4777 nomeFile

Come si può notare la notazione numerica sovrascrive interamente i permessi precedenti, al contrario utilizzando la notazione simbolica è possibile concedere o negare specifici permessi a partire da quelli già assegnati.
Gli utenti vengono rappresentati nella seguente maniera:

Tipologia di Utente	Rappresentazione
Proprietario (Owner / User)	u
Gruppo (Group)	g
Altri (Others)	o
Tutti (All)	a

I permessi sono rappresentati da un carattere:

Permesso	Carattere
Lettura	r
Scrittura	w
Esecuzione	x
Esecuzione Speciale	X
Sticky	t
Setuid/Setgid	s

È possibile concedere permessi aggiuntivi interponendo tra i caratteri che rappresentano gli utenti[6] e i caratteri che rappresentano i permessi aggiuntivi desiderati[7] il carattere '+', rimuovere permessi specifici con il carattere '-' o impostare specifici permessi con il carattere '='.
Il carattere ',' permette di concatenare modifiche a permessi di più tipologie di utenti all' interno dello stesso comando consentendo modifiche diverse ai permessi di ognuno di essi[8].

[6] Ad esempio 'u' per il proprietario, oppure "go" per gruppo e altri.
[7] Ad esempio "rwx", oppure "w".
[8] Ad esempio "u=r,g+x,o-rwx".

Si possono quindi concedere i permessi di lettura, scrittura ed esecuzione al proprietario di un file permettendone lettura al gruppo e agli altri, in maniera analoga a quanto fatto utilizzando la notazione numerica precedentemente, con il comando:

$ chmod u=rwx,g=r,o=r nomeFile

ma si può anche concedere il permesso di esecuzione agli altri, lasciando invariati i permessi preesistenti con il comando:

$ chmod o+x nomeFile

si può rimuovere il permesso di esecuzione concesso agli altri con il comando:

$ chmod o-x nomeFile

si noti che il comando:

$ chmod +s nomeFile

concederebbe sia il permesso Setuid che il permesso Setgid, se si desidera concedere solo il Setuid va utilizzato “u+s”, mentre per il Setgid è richiesto “g+s”.

Il permesso Esecuzione speciale assegna il permesso di esecuzione ad un file solo se è già stato concesso ad almeno uno tra il proprietario, il gruppo e gli altri.

2.3 Accesso a file system FAT e NTFS dal sistema operativo Linux

In questa sezione verrà illustrato come permettere l' accesso in lettura e, volendo, in scrittura a dati memorizzati in file system di tipo File Allocation Table (FAT) e New Technology File System (NTFS) in Linux.
Per raggiungere tale obiettivo si provvederà a modificare il file /etc/fstab applicando quanto descritto nella sezione precedente.
La fonte principale di questa sezione è la voce del manuale relativa a fstab, accessibile online [17] o tramite il comando:

$ man fstab[9]

La definizione che in esso viene fornita è la seguente[10]:
"Il file fstab contiene informazioni descrittive sui vari file system, non viene scritto ma solamente letto dai programmi e la sua creazione e manutenzione sono compiti dell'amministratore di sistema.
Ogni file system è descritto su una sola riga i cui campi sono separati da tabulazioni o da spazi.
L'ordine dei record in fstab è importante, in quanto fsck(8), mount(8), e umount(8) iterano sequenzialmente su questo file per svolgere il loro lavoro."

2.3.1 Configurazione del file /etc/fstab

Le righe che incominciano con il carattere '#' sono commenti, ad eccezione di queste ogni altra riga deve rispettare la struttura:

<file system> <mount point> <type> <options> <dump> <pass>

Viene ora presentato il significato di ciascuno di questi campi, è possibile reperire ulteriori informazioni consultando [17].

- <file system> "definisce il dispositivo speciale a blocchi o il file system remoto da montare, invece di indicare esplicitamente il dispositivo è possibile indicare il file system (ext2 o ext3) di cui effettuare il montaggio[11] tramite il suo UUID o tramite l'etichetta di volume.

[9] sono necessari i pacchetti manpages per la versione in lingua inglese e manpages-it per la versione tradotta in italiano.
[10] Le citazioni riportate in questa sezione sono lievemente rielaborate per permetterne l' inserimento nel presente documento.
[11] effettuato tramite il comando "mount".

Il sistema così sarà più robusto, infatti aggiungere o togliere un disco SCSI cambierà il nome del dispositivo del disco, ma non l'etichetta di volume del file system."
Se si vuole inserire il UUID del file system al posto del suo nome è possibile reperire quest' ultimo con il comando:

blkid

- <mount point> "definisce il punto di mount per il file system."
  Il punto di mount è una directory dalla quale è possibile accedere al contenuto del file system, l' operazione che lega il file system al punto di mount è detta montaggio.

- <type> "definisce il tipo di file system."

- <options> "definisce le opzioni di montaggio associate al file system."
  Nel seguito del capitolo verranno presentate quelle di maggiore rilevanza, ulteriori informazioni sono reperibili nel manuale di mount consultabile tramite il comando:

  $ man 8 mount

- <dump> definisce di quali file system dovrà essere effettuato un backup tramite il comando dump.
  Se il quinto campo non è presente viene restituito il valore zero e il backup non verrà effettuato.

- <pass> "viene usato dal programma fsck(8) per determinare l'ordine di verifica dei file system in fase di riavvio." Se il sesto campo non è presente o è pari a zero, verrà restituito zero e non sarà effettuata alcuna verifica su tale file system da fsck.

Per eseguire un controllo approfondito su quali utenti possano accedere al file system e su quali operazioni possano compiere sui file e sulle directory in esso contenute, si procederà alla creazione di un gruppo a cui verranno aggiunti tutti gli utenti a cui si vogliono concedere gli stessi permessi.[12]
Successivamente si configurerà il file /etc/fstab specificando quali permessi vadano assegnati al gruppo e su quali file system.
Si procede creando il punto di mount e il gruppo:

mkdir <punto di mount>
addgroup <gruppo>

si aggiungono gli utenti al gruppo appena creato:

[12]Si noti che è possibile utilizzare lo stesso gruppo per più file system.

adduser <utente> <gruppo>

si modifica il file /etc/fstab definendo i permessi che si vogliono assegnare a tale gruppo per uno o più file system, si può utilizzare qualunque editor di testo, nell' esempio si è utilizzato gedit:

gedit /etc/fstab

Se il file system è di tipo FAT la riga da aggiungere sarà:

<file system> <mount point> vfat gid=<gruppo>,<permessi> 0 0

mentre se il file system è di tipo NTFS si può aggiungere, assicurandosi di avere installato ntfs-3g [17][13]:

<file system> <mount point> ntfs-3g gid=<gruppo>,<permessi> 0 0

Nel seguito verrà specificato come definire i <permessi>.
Bisogna consentire agli utenti facenti parte del <gruppo> di effettuare il montaggio (mount) e lo smontaggio (umount) del file system tramite il comando sudo, infatti tali comandi richiedono i privilegi di amministrazione per essere eseguiti.
Si procede quindi a modificare il file /etc/sudoers con il comando visudo:

visudo

vanno aggiunte le seguenti righe:

%<gruppo> ALL=/bin/mount <mount point>
%<gruppo> ALL=/bin/umount <mount point>

Una verifica che tutto funzioni correttamente può essere effettuata autenticandosi come un utente appartenente al gruppo <gruppo>, effettuando il mount del file system tramite il comando sudo e controllando i permessi dei file e delle directory in esso memorizzati::

$ sudo mount <mount point>
$ ls -l <mount point>

[13]La maggior parte delle distribuzioni Linux dispongono di un pacchetto precompilato.

Per chiarezza viene presentato un esempio relativo alla procedura appena descritta.
In tale esempio:

<file system> sarà "/dev/sda5"
<mount point> sarà "/media/disco1"
<gruppo> sarà "fswin"
<utente> sarà "utente1"
<permessi> saranno "umask=007"

Si creano il mount point e il gruppo:

mkdir /media/disco1
addgroup fswin

Al gruppo "fswin" viene aggiunto l' utente "utente1" a cui si desiderano concedere i <permessi> per l' accesso al file system con il comando:

adduser utente1 fswin

Si modifica il file /etc/fstab:

gedit /etc/fstab

aggiungendovi:

/dev/sda5 /media/disco1 vfat gid=fswin,umask=007 0 0

se il file system è di tipo FAT.

/dev/sda5 /media/disco1 ntfs-3g gid=fswin,umask=007 0 0

se il file system è di tipo NTFS.

I permessi concessi con "umask=007" concedono al proprietario e al gruppo "fswin" i permessi di lettura, scrittura ed esecuzione dei file contenuti nel file system, mentre tali permessi vengono negati agli altri utenti.
Si concede quindi la possibilità di effettuare il montaggio e lo smontaggio del file system a tutti gli utenti del gruppo "fswin", modificando il file /etc/sudoers con il comando visudo:

visudo[14]

aggiungendovi le seguenti righe:

%fswin ALL=/bin/mount /media/disco1
%fswin ALL=/bin/umount /media/disco1

Si può verificare che la configurazione funzioni correttamente autenticandosi con l' utente "utente1" che è stato in precedenza aggiunto al gruppo "fswin" e utilizzando i seguenti comandi per effettuare il mount e la visualizzazione dei permessi dei file sul file system:

$ sudo mount /media/disco1[15]
$ ls -l /media/disco1/

Le directory risulteranno avere i permessi "drwxrwx—" e i file "-rwxrwx—".
Si noti che i permessi assegnati hanno il seguente significato:
il primo carattere assume valore 'd' ad indicare che il file è una directory o '-' per indicare che è regular file.
Per i file "rwx" corrisponde ai permessi di lettura (read), scrittura (write) ed esecuzione (execute) ed è il permesso concesso sia al proprietario e agli utenti del gruppo "fswin".
Per le directory "rwx" corrisponde alla possibilità di elencare i file e le directory in essa contenuti (r), creare o cancellare file (w), attraversare la directory (x).
"—" corrisponde all' assenza di permessi per gli altri utenti.
Maggiori dettagli sono forniti nella sezione 2.2.

Sono possibili anche altre configurazioni, nel seguito verrà illustrato come impostare i permessi desiderati sui file utilizzando le opzioni umask, dmask e fmask.
I manuali di mount(8) e di ntfs-3g riferiscono che:[16]

- umask=value: Imposta la umask, cioè la bitmask dei permessi che non sono presenti, al valore ottale fornito.
  Il valore predefinito per file system di tipo FAT è la umask del processo corrente.
  Per file system di tipo NTFS il valore predefinito è invece 0, che pertanto garantisce tutti i permessi a tutti gli utenti.

[14] È possibile scegliere un editor di testo differente, ad esempio per usare "nano" è sufficiente il comando:
#EDITOR=nano visudo

[15] Si noti che se al posto di /media/disco1 si scrivesse /media/disco1/ verrebbe riportato un messaggio di errore, bisogna inserire esattamente quanto è indicato nel file /etc/sudoers.

[16] Accessibili con il comando:
$ man 8 mount
$ man ntfs-3g

- dmask=value: Imposta la umask al valore ottale fornito applicandola solamente alle directory.
  Il valore predefinito è la umask del processo corrente per file system di tipo FAT e 0 per file system di tipo NTFS.

- fmask=value Imposta la umask al valore ottale fornito applicandola solamente ai regular file.
  Il valore predefinito è la umask del processo corrente per file system di tipo FAT e 0 per file system di tipo NTFS.

Vengono ora presentati alcuni esempi volti a mostrare i possibili utilizzi di tali opzioni, il primo ha come obiettivo quello di consentire l' accesso in lettura e scrittura ai dati contenuti in un file system al proprietario "utente1", in lettura al gruppo "fswin" e nessun permesso agli altri.
Per ottenere tale risultato è necessario aggiungere al file /etc/fstab la seguente riga se il file system è di tipo FAT:

<file system> <mount point> vfat auto,uid=utente1,gid=fswin,dmask=027,fmask=137 0 0

o la seguente se è di tipo NTFS:

<file system> <mount point> ntfs-3g auto,uid=utente1,gid=fswin,dmask=027,fmask=137 0 0

Il significato di tali opzioni è il seguente:

- auto: Permette di effettuare il mount del file system utilizzando il comando "mount -a".
  Essendo tale comando usato all' avvio permette anche di ottenere il mount in automatico.

- uid=utente1: uid [17] specifica l' utente proprietario del file system, in questo caso "utente1".
  È anche possibile utilizzare l' uid numerico associato ad un utente, tale valore è reperibile analizzando il file /etc/passwd, per esempio uid=1001.

- gid=fswin: gid[18] specifica il gruppo del file system, in questo caso "fswin".

[17] Acronimo di "user identifier".
[18] Acronimo di "group identifier".

È anche possibile utilizzare il gid numerico ad esso associato, quest' ultima informazione è reperibile analizzando il file /etc/group.

- dmask=027: dmask permette di specificare quali permessi si desidera vengano negati relativamente alle directory.
  Nell' esempio, utilizzando 027, si vogliono negare il permesso di scrittura al gruppo e tutti i permessi agli altri.
  Questo valore viene sottratto dal massimo valore ammesso (777), il risultato sarà di avere per le directory di questo file system i permessi: 777 – 027 = 750, garantendo quindi per il proprietario i permessi di lettura, scrittura ed esecuzione, per il gruppo permessi di lettura ed esecuzione e negando ogni permesso agli altri.

- fmask=137: fmask permette di specificare quali permessi si desidera vengano negati relativamente ai file, come per dmask e umask tali permessi vengono sottratti dal massimo valore ammesso (777).
  Nell' esempio, utilizzando il valore 137, si concederanno i seguenti permessi: 777 – 137 = 640.
  Si concedono quindi i permessi di lettura e scrittura al proprietario, il permesso di lettura al gruppo e si negano tutti i permessi agli altri.

È possibile anche specificare permessi comuni sia per i file che per le directory utilizzando l' opzione umask, per esempio aggiungere la riga:

<file system> <mount point> <vfat / ntfs-3g> umask=000 0 0

concede a tutti gli utenti i permessi di lettura, scrittura ed esecuzione.
Un altro esempio possibile può essere concedere al proprietario e al gruppo i permessi di lettura e scrittura sui file[19]:
<file system> <mount point> <vfat / ntfs-3g> gid=fswin,dmask=007,fmask=117 0 0

Con le configurazioni presentate il montaggio e lo smontaggio dei file system sono consentite solamente all' utente root o, se è stata concessa l' autorizzazione modificando il file sudoers tramite il comando visudo, anche ad utenti che utilizzando il comando sudo abbiano ottenuto i privilegi di amministrazione.
L' aggiunta dell' opzione "users" ad una riga di /etc/fstab relativa ad un file system FAT, permette il montaggio e lo smontaggio di tale file system anche a utenti normali, in tal caso essi vengono associati come proprietario del file system fino allo smontaggio, mentre se aggiunta ad una riga relativa ad un file system NTFS ne consente solo lo smontaggio.

[19]Per le directory è necessario concedere il permesso di esecuzione altrimenti risultano inutilizzabili, si veda la sezione 2.2

2.4 Accedere a dati memorizzati in file system di tipo ext2/ext3 da Microsoft Windows

Ext2 ed ext3 attualmente sono i file system maggiormente utilizzati dalle distribuzioni Linux.
In questa sezione verrà illustrato come sia possibile accedere sia in lettura che in scrittura a tali file system dai sistemi operativi Microsoft Windows.
Si continua pertanto a mostrare l' approccio in cui dei dati venga mantenuta una singola replica, memorizzata in un file system a cui viene poi effettuato l' accesso da diversi sistemi operativi.

2.4.1 Ext2/Ext3

Viene ora illustrata la configurazione e l' utilizzo dell' applicazione Ext2Fsd [18].
Ext2Fsd, di cui attualmente è disponibile la versione 0.48, è un driver per file system ext2/ext3 che permette di associare una lettera su Microsoft Windows a questa tipologia di file system e di accedere in lettura e scrittura ai dati in esso memorizzati.
Da [18] è possibile scaricarne l' installer, durante l' installazione vengono concesssa la possibilità di attivare il supporto alla scrittura con le opzioni "enable write support for Ext2 partitions" e "enable force writing support on ext3 partitions" e viene richiesto se si desidera avviarlo automaticamente all' avvio "make Ext2Fsd automatically started when system boots".
Al termine dell' installazione eseguendo "Ext2 Volume Manager" (Ext2Mgr) sarà possibile assegnare una lettera ai file system a cui si desidera accedere ed essi saranno accessibili come qualunque volume di cui Microsoft Windows fornisca il supporto nativamente, è sufficiente selezionare "Change Drive Letter" → "Add..." dopo avere cliccato col tasto destro sul file system desiderato.
È necessario scegliere uno tra i tre metodi che l' applicazione fornisce per effettuarne il montaggio, il testo seguente è una traduzione adattata di [19]:

- Windows API DefineDosDevice è comodo per un uso temporaneo, la lettera associata al drive durerà solo fino al primo riavvio.

- Windows MountMgr + Ext2Mgr permette il mount automatico quando viene rilevata l' aggiunta di un disco e la rimozione della lettera quando un disco viene rimosso.
  È molto utile quando si utilizzano dischi removibili.

- Memorizzare nel registro di Windows la lettera permette di mantenerla anche dopo un riavvio.
  È molto utile per dischi non removibili.

Per un uso occasionale si può scegliere di non avviare automaticamente Ext2Fsd all' avvio di Windows, ma di utilizzare il secondo metodo di mount,[20] in questo modo il mount non sarà eseguito in automatico ma solo all' avvio di Ext2Mgr.
In caso le modifiche non siano immediatamente visibili è possibile forzarne il rilevamento con "Tools" → "Reload and Refresh".
Per un uso intensivo si può invece scegliere la terza opzione "Create a permanent mount point via Session Manager", in tal caso risulta necessario un riavvio del computer.
È inoltre possibile effettuare il mount in modalità "sola lettura", tale opzione è accessibile dal tasto destro del mouse alla voce "Ext2 Management" selezionando "Mount volume in readonly mode".

Risoluzione Problemi

Se l' applicazione dovesse segnalare il messaggio di errore "Il disco nell'unità X non è formattato, formattare il disco?" o impedisse di associare una lettera, il problema potrebbe risiedere nel mancato avvio del servizio di Ext2fsd.
Per assicurarsene si deve visionare "Tools" → "Service Management", se appare il messaggio "Ext2Fsd is NOT started" è possibile avviare manualmente il servizio con il pulsante "start".
Se si sta utilizzando il sistema operativo Windows Vista, si ricorda che è necessario eseguire sia l' installar che l' applicazione Ext2Mgr con privilegi elevati.
Al momento l' applicazione non è disponibile per Windows 7.

2.4.2 Ext4:

Ext4 è un file system che è stato solo recentemente adottato come scelta predefinita dalla maggior parte delle distribuzioni Linux più diffuse.
Fornisce numerose funzionalità aggiuntive rispetto al suo predecessore, tra queste vi sono gli extent che permettono di aumentare le prestazioni del file system e di ridurne la frammentazione. [20]
Proprio attivando tale funzionalità si perde la compatibilità con le applicazioni che permettono di accedere ai dati memorizzati in file system di tipo ext2 ed ext3.

[20] "Automatic mount via MountMgr" che risulta essere la scelta predefinita.

Bibliografia

[1] http://www.cis.upenn.edu/~bcpierce/unison/
[2] http://www.cis.upenn.edu/~bcpierce/unison/download/releases/beta/unison-manual.html
[3] T. Jim, B. C. Pierce, and J. Vouillon. How to build a file synchronizer.
[4] B. C. Pierce, J. Vouillon. "What is Unison? A Formal Specification and Reference Implementation of a File Synchronizer"
[5] http://alan.petitepomme.net/unison/index.html
[6] http://sourceforge.net/projects/pidgin/files/GTK%2B%20for%20Windows/2.14.7.0/gtk-runtime-2.14.7.0.zip/download
[7] http://www.microsoft.com/downloads/details.aspx?familyid=9B2DA534-3E03-4391-8A4D-074B9F2BC1BF&displaylang=en
[8] http://alan.petitepomme.net/unison/assets/2010.04.21-ESup-Unison-Unison_GTK_2.40.16.zip
[9] http://www.cygwin.com/
[10] http://www.openssh.com/
[11] http://www.cygwin.com/cygwin-ug-net/using-cygwinenv.html
[12] http://www.cygwin.com/cygwin-ug-net/using-utils.html
[13] http://www.seas.upenn.edu/~bcpierce/unison//download/releases/beta/
[14] http://www.cis.upenn.edu/~bcpierce/unison/download/releases/beta/unison-manual.html#usingmultiple
[15] K.K. Mookhey, Nilesh Burghate and ISACA. Linux - Security, Audit and Control Features
[16] Carla Schroder. Linux cookbook
[17] http://manpages.debian.net/cgi-bin/man.cgi?query=fstab&apropos=0&sektion=0&manpath=Debian+5.0+lenny&format=html&locale=it
[18] http://www.ext2fsd.com/
[19] http://www.ext2fsd.com/?page_id=7
[20] https://ext4.wiki.kernel.org/index.php/Ext4_Howto#Extents