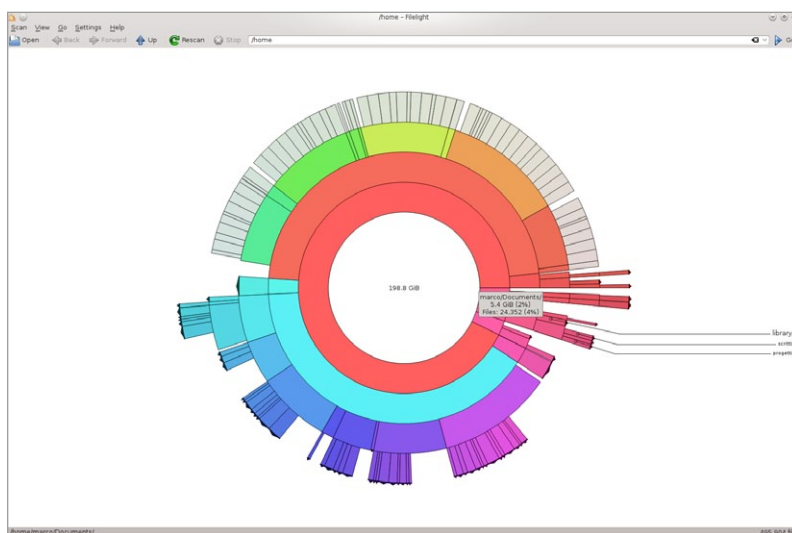


Come funzionano i file system su Linux

Dischi fissi, Nas, chiavi Usb: come viene gestito a basso livello lo storage dal sistema operativo Open Source

Installazione e riconoscimento hardware in distribuzioni Gnu/Linux come Ubuntu, Mint o Fedora sono molto migliorati negli ultimi anni. Nella grande maggioranza dei casi bastano un Dvd, un po' di pazienza e una serie di clic che non richiedono conoscenze particolari. Almeno finché non ci sono troppi dischi da controllare. Oggi però ai dischi rigidi interni si aggiungono sempre più frequentemente schiere di chiavi Usb e diversi dischi esterni di enormi dimensioni. Anche le periferiche Nas (*Network Attached Storage*) che permettono di condividere Terabyte di dati su qualsiasi rete locale, anche domestica, diventano sempre più invitant. Gestire nel modo migliore questa enorme quantità di spazio crea problemi nuovi, soprattutto quando gran parte di esso è in computer o dischi *portatili*, quindi a maggior rischio di furto. Per affrontarli è utile, se non indispensabile, saperne qualcosa di come viene gestito lo storage. Questo è proprio ciò che cercheremo di fare, per quanto riguarda Linux e il software Open Source in generale, in questo e nel prossimo numero della rubrica. Soffermiamoci ora su quegli strumenti di base che, dietro le quinte di ogni sistema Linux, gestiscono a basso livello file, cartelle e partizioni. Nel prossimo numero vedremo alcuni trucchi e funzioni più avanzate nello stesso campo, oltre a un argomento già affrontato qualche anno fa: il backup degli stessi sistemi con gli strumenti Open Source più moderni.



Quali parti di Linux occupano più spazio su disco, e perché? Quali sarebbe più opportuno muovere su una partizione separata? Utility come FileLight aiutano a rispondere a queste e altre domande.

Sotto il cofano di Linux: partizioni, file system, blocchi e inode

Una delle prime cose che si imparano quando si cerca di capire come funzionano i computer è che un dispositivo permanente di storage va organizzato in una o più zone semi-indipendenti, chiamate *partizioni*. A meno di configurazioni particolari, di cui ripareremo, una partizione è una sorta di compartimento stagno di uno qualsiasi di quei dispositivi. Quello che un programma fa all'interno di una certa partizione (incluse attività potenzialmente devastanti come scrivere dati in uno o più file, senza mai fermarsi), non si può propagare ad altre partizioni.

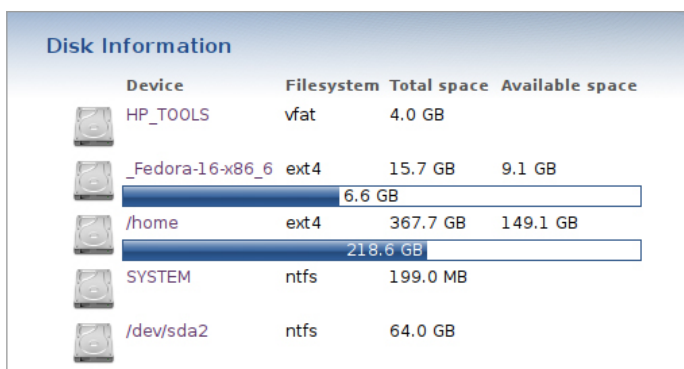
Oltre alla robustezza, utilizzare diverse partizioni contribuisce ad aumentare la sicurezza di un computer sistema, poiché rende più facile regolare nella maniera

migliore l'accesso a ogni sua parte. Per esempio, se tutti i programmi e i relativi file di configurazione generali (non quelli aggiuntivi di ogni utente) sono l'unico contenuto di una sola partizione, vi si può inibire qualsiasi operazione di scrittura al di fuori degli aggiornamenti: in quel modo il computer funzionerà come al solito ma sarà impossibile a chiunque, si tratti di malware o utenti distratti, cancellare file essenziali.

Su computer Linux particolarmente carichi cambiare schema di partizionamento potrebbe migliorare anche le prestazioni, sia pure di poco. C'è chi, per esempio, dispone dati e cartelle usati più di frequente, per non parlare della partizione di *swap*, nei settori di un disco che hanno tempi d'accesso medi minori.

In Linux esistono due tipi principali di partizioni: la *swap* appena nominata, che costituisce una zona d'appoggio alla Ram, e quelle che contengono i

Oltre a programmi come File Light, i desktop KDE e Gnome offrono visioni d'insieme di tutte le periferiche di storage tramite interfacce dedicate.



Device	Filesystem	Total space	Available space
HP_TOOLS	vfat	4.0 GB	
_Fedora-16-x86_6	ext4	15.7 GB	9.1 GB
/home	ext4	367.7 GB	149.1 GB
SYSTEM	ntfs	199.0 MB	
/dev/sda2	ntfs	64.0 GB	

dati. In questa seconda categoria sono incluse anche partizioni speciali, che non contengono file ordinari. Su Linux, per esempio, si adotta spesso una partizione separata per la cartella */boot*, perché essa non solo va gestita e protetta in maniera particolare, ma dev'essere disponibile fin dai primissimi momenti dell'avvio del sistema. È infatti in */boot* che risiedono il kernel Linux vero e proprio, i suoi file di configurazione e il boot loader. Quest'ultimo, chiamato Grub2, è il programma che consente a più versioni di Linux e Windows di coesistere nello stesso computer, e all'utente di scegliere quale lanciare a ogni accensione. Le altre parti di un sistema Linux che più spesso conviene porre su partizioni indipendenti sono */home*, */opt* e, soprattutto su sistemi embedded, alcune sottocartelle di */var*. La prima è infatti il luogo dove risiedono i file personali di ogni utente, ognuno nella sua cartella. */opt* è, secondo gli standard Linux, il luogo dove installare programmi opzionali di terze parti, che non hanno nulla a che vedere con la distribuzione in quanto

tale. Nelle reti locali grandi e piccole sia */home* sia */opt* risiedono fisicamente su un server separato e sono montate (nel senso spiegato nei paragrafi successivi) da ogni computer della rete locale, per semplificare manutenzione e backup.

Come si passa da partizioni a cartelle?

Le interfacce grafiche con cui accediamo ai file in un computer possono variare parecchio nell'aspetto esteriore, soprattutto in questi tempi di tablet e cloud computing, ma la loro architettura generale è sempre la stessa. I file sembrano vivere in cartelle organizzate più o meno gerarchicamente oppure, nei desktop semantici come KDE/Nepomuk su Linux, associati a varie etichette. Come si arriva a questo, partendo dai byte? Per prima cosa, serve un *block device*, termine traducibile più o meno con "dispositivo a blocchi". In generale, sotto Linux i dispositivi di storage e input/output possono essere a caratteri

oppure a blocchi. I primi sono ad accesso sequenziale, nel senso che i caratteri o byte vi si possono leggere solo uno alla volta, nella sequenza in cui erano sul dispositivo o vengono generati in tempo reale. Il dispositivo a carattere più semplice è un qualsiasi terminale con riga di comando. I dispositivi a blocchi sono invece ad accesso casuale: da essi si può leggere qualsiasi byte in qualsiasi momento, non importa in quale punto si trovi o quando ci sia arrivato.

Da quanto detto si capisce che, con l'eccezione dei backup, desktop o server Linux possono tenere file e cartelle su qualsiasi periferica, basta che si comporti come un dispositivo a blocchi. Solo in quel caso è infatti possibile definire partizioni che potranno a loro volta ospitare un file system, e quindi file sempre accessibili, secondo le esigenze del momento.

Cosa è esattamente un file system? In breve, si tratta di una struttura dati predefinita, che specifica in quale formato e ordine i singoli file e i relativi metadati possono essere disposti su disco, e regola l'accesso agli stessi oggetti.

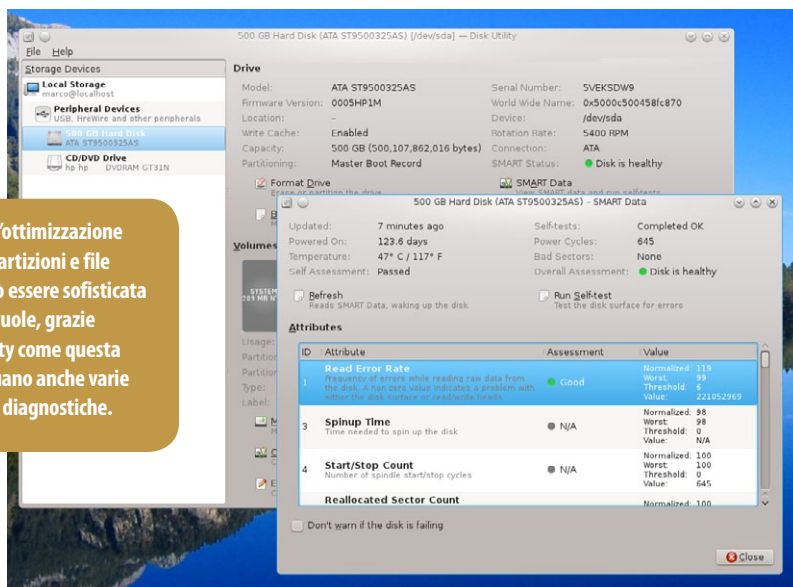
I driver dei file system attuali effettuano anche il *journaling*. Questa operazione consiste nel tenere un diario (in inglese *journal*) di tutte le modifiche più recenti apportate al file system, cioè a file e cartelle sul disco. È grazie a quel diario che i riavvii di Linux, anche dopo interruzioni di corrente, disconnessioni improvvise di dischi esterni o altri eventi traumatici, sono molto più rapidi di quanto erano una volta. Un file system con *journaling*, comunque, non può proteggere da guasti hardware veri e propri oppure da bachi nel software.

Linux e i file system di Windows

Linux può leggere e scrivere senza particolari problemi i principali file system creati per Dos prima e per Windows dopo, da Fat a Ntfs. L'ultimo richiede accorgimenti particolari come l'uso del sistema Fuse (*File System in Userspace*, <http://fuse.sourceforge.net/>), di cui parleremo nel prossimo numero, e l'installazione del driver Ntfs-3g (www.ntfs-3g.com), ma non è più un problema come una volta. Ntfs-3g supporta tutte le operazioni basilari sui file, oltre a compressione trasparente e liste di controllo accessi. Quello che ancora manca è il supporto completo per la cifratura e il *journaling* di Ntfs. I file system Microsoft più vecchi, come Fat16 e Fat32 con o senza estensioni Vfat, possono ancora essere utili su chiavi Usb o dischi esterni, se si vuole avere la certezza che saranno accessibili da qualsiasi sistema operativo. Linux supporta questi file system senza problemi, ma non bisogna mai dimenticare le loro limitazioni. Altrimenti si potrebbero

avere spiacevoli sorprese, soprattutto se vengono usati per backup di partizioni Linux. Il limite più visibile è quello sulle dimensioni (massimo 2 GByte sui più vecchi) e ancor più sui nomi dei file. Senza le estensioni Vfat, che reggono nomi fino a 255 caratteri, spazi e più punti, i file system Fat sono ancora fermi ai tempi del Dos, cioè a nomi con massimo di otto caratteri e una sola estensione di non più di tre lettere. Inoltre i file system Fat, soprattutto Fat16, sprecano parecchio spazio, perché in essi i file occupano numeri interi di *cluster*. La dimensione minima di queste zone in cui viene diviso il disco rigido è 4kByte in Fat32 e addirittura 32 in Fat16. Nel secondo caso, un file di dimensione pari a 32 KByte più un Byte occuperà 64 KByte! L'ovvia conseguenza è che, se per avere la massima portabilità si fanno backup di partizioni Linux su un disco partizionato Fat, quest'ultimo dovrebbe essere di dimensioni sensibilmente superiori a quelle della cartella con dentro gli stessi file in un file system Linux.

L'analisi e l'ottimizzazione di dischi, partizioni e file system può essere sofisticata quanto si vuole, grazie a Disk Utility come questa che effettuano anche varie operazioni diagnostiche.



Niente lettere su Linux, solo punti di montaggio

Aver formattato una partizione con un file system non significa ancora, almeno su Linux, poterla usare. Prima bisogna montarla. Avete notato che in Linux non esistono lettere che distinguano una partizione dall'altra, come C: o A: ? Il motivo è che in Linux tutte le partizioni, o meglio i file system che contengono, vengono connesse a un unico albero di cartelle e sottocartelle che inizia con la cartella `/root`. L'associazione di una partizione a una cartella è proprio quello che viene chiamato *montaggio*: è durante quell'operazione che vengono definiti i permessi d'accesso, la possibilità di modificare il contenuto e tutti gli parametri di un file system.

Inode e file. Non serve altro

Un file system rappresenta i file come inode. Gli inode sono strutture dati identificate da un numero di serie unico (a livello di partizione), che contiene informazioni sul file come proprietario, permessi, date e ore di creazione, modifica e ultimo accesso e, ovviamente, l'effettiva posizione dei suoi byte sul disco. Praticamente, le uniche informazioni su un file non scritte nel suo inode sono il suo nome e cartella d'appartenenza, per motivi che saranno chiari fra un attimo. Quando una partizione viene inizializzata vi si crea un numero fisso di inode. Questo è il massimo numero di oggetti di ogni tipo (comprese directory, file speciali, link) che potranno esistere in ogni momento

su quella partizione.

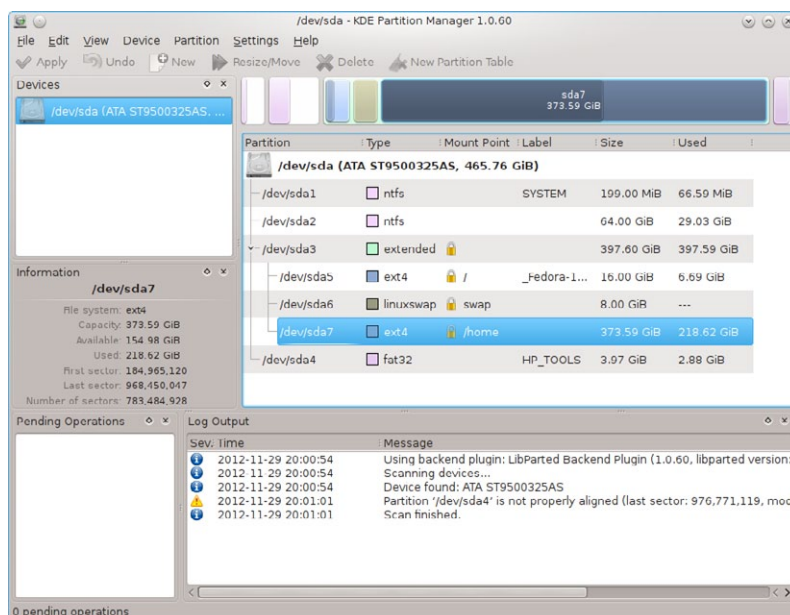
Avere inode con cui gestire file è sufficiente perché da sempre in Unix, di cui Linux continua a essere l'incarnazione Open Source più popolare, a basso livello tutto è un file. Anche quello che per noi utenti non lo è, da directory e periferiche hardware a connessioni dirette fra più programmi locali o remoti. Stampare un file consiste nello scriverne il testo, nel file che rappresenta la stampante e il suo driver, nello stesso modo in cui lo si scriverebbe in un altro file normale. Il kernel tiene traccia dei file utilizzando indici chiamati *in-core inode*. I file che rappresentano periferiche hardware sono in directory riservate, come `/dev`. Qualunque sia il loro contenuto,

le directory in Linux sono sempre file, che però contengono nomi e posizioni di tutti i file al loro interno. È grazie a questa struttura che uno stesso file può, fra le altre cose, avere diversi nomi ed essere utilizzato contemporaneamente da diversi processi.

I File System di Linux

Oggi come oggi, a meno che il proprietario non voglia sperimentare o abbia esigenze particolari, dentro o intorno un sistema Linux si trovano soltanto file system della serie Ext. La versione attuale, quella usata per default negli installer di parecchie distribuzioni, è Ext4. I suoi predecessori, Ext2 ed Ext3, sono comunque ancora piuttosto diffusi. In alcuni casi, la ragione è semplice inerzia, o trascuratezza, degli utenti. In altri ci sono ragioni tecniche molto valide. Ext2, per esempio, è preferito in molti casi in cui si devono usare unità di storage Flash, che tollerano un numero relativamente basso di scritture prima di guastarsi. Ext2 può prolungarne la vita perché, essendo più semplice e limitato, ma soprattutto privo di *journaling*, effettua meno scritture degli altri. Il *journaling* è arrivato in Linux solo con Ext3. Oggi, in Ext4, lo si può disattivare sia per prolungare la vita dei dischi Ssd sia per aumentare (di poco) le prestazioni quando ogni millisecondo è importante. Dal punto di vista delle dimensioni, Ext4 supporta file system fino a 1 ExaByte e singoli file fino a 16 TeraByte. Inoltre, a differenza di Ext3 che si fermava a 32000, Ext4 può gestire

Creare e riconfigurare partizioni non è più un'operazione per super esperti, nemmeno su Linux. Sia Kde (nell'illustrazione) sia Gnome hanno dei Partition Manager completi e facili da usare.



qualsiasi numero di sottodirectory in ogni directory. Il problema è che questi numeri, per quanto grandi possano sembrare, già non bastano più ad alcuni utenti.

Il futuro è btrfs?

Per ora Ext4 è il file system predominante sui sistemi Linux aggiornati, ma c'è già chi guarda avanti per risolvere nuovi problemi. Il principale è la necessità di gestire sistemi di archiviazione di grandissime dimensioni che stanno diventando comuni nei data center di oggi. Questo è un problema che non si risolve semplicemente aumentando la dimensione massima dell'intero file system o di ogni suo singolo file. Per lavorare in quelle condizioni servono anche nuove capacità di rilevare, tollerare e riparare errori con la massima velocità possibile, nonché backup molto più efficienti di quelli attuali. Al momento la risposta più promettente a queste esigenze, anche se non ancora matura, sembra essere Btrfs (B-tree file system, <https://btrfs.wiki.kernel.org>). Questo nuovo modello, sviluppato da una comunità che include fra gli altri pezzi da novanta dell'informatica come Oracle, Fujitsu, Intel, Red Hat e SUSE, ha obiettivi davvero ambiziosi. Btrfs dovrebbe essere molto più resistente ai guasti dei suoi predecessori, senza essere troppo complicato da installare, gestire e riparare. Allo stesso tempo, il nuovo file system sarà ottimizzato per la *cloud computing*, cioè per sistemi di storage remoti, gestiti da terzi e di dimensioni variabili in tempo reale o quasi, secondo le esigenze dell'utente. Per quanto concerne la robustezza, Btrfs sarà capace, fra le altre cose, di *checksum* e *snapshot*. Le prime sono operazioni matematiche con cui si può verificare velocemente se un file o insieme di file è stato modificato. I secondi sono paragonabili a fotografie istantanee, che rappresentano lo stato completo del file system in un determinato momento. Avere snapshot è utilissimo, per non dire indispensabile, per fare backup consistenti senza interrompere l'attività dei programmi che usano il file system. Queste caratteristiche sono cruciali per aumentare l'utilità e la diffusione di Linux nei grandi data center. Nonostante questo Btrfs non trascura gli utenti di normali desktop o media center da salotto. La lista delle funzioni da sviluppare include infatti una gestione molto più efficiente anche di file di piccole dimensioni e dischi Ssd. Come

Strumenti per cifratura di file e partizioni sotto Linux



La cifratura è un'attività complessa, non solo e non tanto per gli algoritmi usati dai programmi corrispondenti, che tutto sommato interessano solo ai programmatori. Per tutti gli altri, il problema maggiore è capire davvero come gli conviene usarla personalmente. Questa rassegna, assolutamente incompleta per ragioni di spazio, ha appunto lo scopo di mostrare quanto è varia l'offerta Open Source e qual è il minimo che si dovrebbe sapere con certezza prima di installare qualcosa di questo genere. Leggendo le brevi descrizioni che seguono diventa infatti evidente che, prima di cominciare, si dovrebbe almeno trovare la propria risposta a domande come: cos'è che voglio davvero proteggere? Solo file, oppure anche quello che sto facendo in ogni momento nel mio computer? Quanti e quali file vanno davvero protetti? Tutti o solo alcuni? I file cifrati dovranno essere accessibili da vari sistemi operativi?

DM-CRYPT (www.saout.de/misc/dm-crypt/)

Dm-crypt non è altro che la parte dedicata specificamente alla cifratura del sistema DeviceMapper descritto nell'articolo principale della rubrica. Grazie a questa sua origine, Dm-crypt cifra senza problemi sia swap e partizioni normali sia i volumi Raid o logici di cui parleremo il mese prossimo. Si può inoltre cifrare qualsiasi file system supportato dalla versione di Linux con cui si sta usando Dm-crypt.

LOOP-AES (<http://sourceforge.net/projects/loop-aes/>)

Loop-Aes è un meccanismo di cifratura per Linux veloce e trasparente, una volta installato. Consiste infatti di alcuni moduli del kernel che vanno scaricati e configurati prima dell'uso. Se disponibile, loop-Aes sfrutta anche le funzioni di accelerazione hardware di alcune schede grafiche. L'algoritmo usato è l'Advanced Encryption Standard utilizzato anche dal governo degli Stati Uniti. Loop-Aes può cifrare sia partizioni normali sia la swap del computer, impedendo quindi a chi si sia connesso illecitamente di analizzarne le attività in tempo reale, o di copiare informazioni riservate non disponibili su file ma solo in Ram. Le chiavi di cifratura possono essere conservate in un file, a sua volta cifrato con altre chiavi e algoritmi, per aumentare la sicurezza.

LUKS (<http://code.google.com/p/cryptsetup/>)

Luks è la sigla di Linux Unified Key Setup, un formato standard per la crittografia su disco che usa Dm-crypt come back-end. L'obiettivo di Luks è eliminare qualsiasi potenziale problema di compatibilità fra cifratura e decifratura di una stessa partizione da parte di distribuzioni Linux con librerie o kernel diversi, anche quando si cambia distribuzione ogni volta che si riavvia il Pc. Oltre a descrivere come effettuare la cifratura a basso livello, Luks specifica anche procedure portabili e standard di gestione per le relative chiavi. Il sito ufficiale di Luks ospita anche cryptsetup, una utility per configurare Luks tramite dm-crypt sulla riga di comando. Le partizioni cifrate con Luks sono accessibili anche da Windows e Windows Mobile con il programma FreeOtf (www.freeotf.org).

SCRAMDISK PER LINUX (SD4L, <http://sd4l.sourceforge.net>) E TRUECRYPT (www.truecrypt.org)

Chiudiamo citando insieme ScramDisk e TrueCrypt perché svolgono sostanzialmente le stesse funzioni, più o meno nello stesso modo, anche se solo il primo è realmente Open Source. Sia ScramDisk sia TrueCrypt offrono quella che viene chiamata Crittografia negabile: i loro container sembrano ammassi casuali di byte, quindi a meno di non sapere in anticipo quali e dove sono, non si riesce nemmeno a vederli, e quindi ad attaccarli per cercare di indovinarne la chiave.

SD4L è un sistema di crittografia in tempo reale completo. Nel pacchetto sono compresi un driver del kernel, un'interfaccia grafica e cinque programmi da riga di comando per riformattare, riparare o convertire in varie maniere i dati cifrati. ScramDisk crea dei contenitori, copiabili e riutilizzabili così come sono su altri computer, che possono essere aperti anche con TrueCrypt. Nel verso opposto, ScramDisk sa leggere e riparare (entro certi limiti!) contenitori creati con le versioni 6 e 7 di TrueCrypt. Un singolo contenitore ScramDisk può corrispondere a una singola cartella, a una partizione di swap oppure a un intero disco rigido, anche se quest'ultimo contiene più partizioni. Sotto Linux i contenitori vengono automaticamente smontati, e quindi resi inaccessibili da altri programmi per maggior sicurezza, dopo un periodo di inattività preconfigurabile.

dicevamo, Btrfs non è ancora pronto per l'uso in massa, ma a partire dalla versione 3.7 del kernel (vedi News) dovrebbe essere più facile provarlo anche per gli utenti non programmatori.

Cifratura: come funziona su Linux e quando usarla

Non possiamo non chiudere questa prima parte del viaggio nella gestione dei file sotto Linux citando brevemente come affronta un problema tanto più critico quanto più aumenta l'uso di dischi, laptop e altri terminali mobili: come impedire l'accesso a dati riservati quando il dispositivo che li contiene viene perso o rubato? Linux contiene funzioni di cifratura tali da soddisfare anche gli utenti più esigenti, grazie ai prodotti descritti nell'altro articolo di questo mese, e un'architettura che permette di aggiungere nuovi metodi senza troppi sforzi. Prima dell'arrivo del kernel Linux 2.6 si usavano per la cifratura vari *device* speciali chiamati *loopback*, più o meno incompatibili fra loro. In generale, per ogni algoritmo di cifratura serviva un *loopback device* diverso. A partire dal kernel Linux 2.6 questo meccanismo è stato abbandonato a favore di un altro, chiamato DeviceMapper, che viene usato per gestire snapshot come quelli di Btrfs e i sistemi multivolume o Raid di cui parleremo il mese prossimo. Il DeviceMapper è un filtro generico che, dopo aver creato un Block Device virtuale, elabora i dati che quest'ultimo riceve e li passa ad un altro Block Device. L'operazione funziona in maniera trasparente per utenti e applicazioni anche nella direzione inversa, per la lettura di dati. Quando è utilizzato per la cifratura, DeviceMapper crea un nuovo dispositivo nella directory `/dev/mapper`, che può essere montato come se fosse un normale disco interno o esterno. I dati scritti da a questo dispositivo sono cifrati, con algoritmi come Aes o Blowfish, dal modulo corrispondente di DeviceMapper. È il risultato di questa operazione che poi viene effettivamente salvato nel disco reale. Nonostante i progressi bisogna stare attenti a non strafare. Cifrare i propri dati è un'ottima abitudine, che dovrebbe diventare la regola per tutti. Le distribuzioni più popolari di Linux consentono al momento dell'installazione di cifrare tutta la cartella `/home`, o anche intere partizioni, con pochi clic. Forse è meglio cifrare solo come e dove serve davvero, e soprattutto affrontare seriamente la gestione del sistema che ne risulta.

Linux News

Fuorusciti Nokia resuscitano Meego, per il mercato cinese

Meego (<https://meego.com/>), poi chiamato Mer, è un sistema operativo per terminali mobili basato su Linux e sponsorizzato da Nokia, che poi perse interesse nel progetto. A novembre 2012 la startup finlandese Jolla, formata in buona parte da ex dipendenti di Nokia, ha annunciato che, partendo proprio da Mer, svilupperà un nuovo sistema operativo dello stesso tipo, chiamato Sailfish OS (<https://sailfishos.org>). Il principale supporto al progetto arriverà a Jolla dalla Cina. D.Phon, il più grande rivenditore di terminali mobili di quel paese, vuole infatti avere a disposizione una piattaforma software mobile che non abbia restrizioni di proprietà intellettuale e soprattutto sia completamente indipendente da Android di Google.

Gnome 3.8 fonde elementi tradizionali e nuove applicazioni

Fallback, alla lettera "caduta all'indietro", in informatica indica il ritorno semiautomatico di un sistema o programma software alla sua versione precedente di software. Il tradizionale sistema di fallback all'interfaccia grafica della serie 2 non sarà più presente in Gnome 3.8, che dovrebbe arrivare nelle distribuzioni Linux a inizio 2013. Fortunatamente questa decisione degli sviluppatori non significa affatto che gli utenti a cui non piacciono i recenti cambiamenti di Gnome debbano cercarsi un'altra soluzione. La task bar e il menu applicazioni classici di Gnome 2.x rimarranno infatti disponibili come estensioni caricabili con pochi clic, anziché come rimpiazzi, della Shell Gnome, che è l'interfaccia grafica vera e propria di Gnome 3.8. Questa personalizzazione è in realtà possibile anche con versioni precedenti di Gnome 3, ma finora installarla e configurarla era una procedura interamente a carico dell'utente.

Linux Kernel 3.7, nuove funzioni per server e desktop

La versione 3.7 del programma che è al cuore di qualsiasi ambiente chiamato genericamente "Linux" il cui rilascio è previsto a fine 2012, porterà diverse novità immediatamente apprezzabili dagli utenti finali, quando la installeranno a mano o arriverà con le nuove distribuzioni. Il cambiamento più interessante sarà senz'altro l'inclusione di nuovi driver grafici Open Source, più veloci e completi, per i chipset di tutti i principali fornitori, da nVidia a Intel e Radeon. In particolare, il nuovo driver Nouveau per schede nVidia (<http://nouveau.freedesktop.org>) potrebbe consentire a molti più utenti di fare a meno di quello proprietario ufficiale. Un'altra parte di Linux 3.7 da tenere sott'occhio, per i motivi spiegati nell'articolo principale, è il nuovo driver sperimentale per file system Btrfs.

KDE compie 15 anni e li festeggia con una nuova beta

Nello stesso mese in cui è diventato quindicenne, il desktop Open Source Kde ha rilasciato la prima beta della sua prossima versione, la 4.10. I suoi punti di forza dovrebbero essere nuovi algoritmi di correzione dei colori e un uso estensivo delle librerie grafiche Qt Quick (<http://qt.nokia.com/qtquick/>). Questi componenti, nati per velocizzare lo sviluppo di interfacce grafiche con effetti visivi estremamente fluidi, sono alla base del sistema di blocco dello schermo e degli sfondi animati di Kde 4.10. Dietro le quinte del nuovo desktop dovrebbero esserci anche un nuovo Manager di stampa e un Nepomuk (motore di ricerca semantico per desktop) più veloce delle versioni precedenti, con etichette immediatamente visibili e utilizzabili da qualsiasi applicazione Kde.