

Amministrazione avanzata

Capitolo 12

12. Amministrazione avanzata	pag. 301
1. RAID e LVM	pag. 302
1. Software RAID	pag. 302
1. I diversi livelli RAID	pag. 303
2. Come configurare un RAID	pag. 305
3. Backup della configurazione	pag. 311
2. LVM	pag. 312
1. I Concetti LVM	pag. 313
2. Come configurare un sistema LVM	pag. 313
3. La tecnologia LVM ed il manifestarsi di nuove esigenze nel corso del tempo	pag. 318
3. RAID o LVM?	pag. 320
2. Virtualizzazione	pag. 323
1. Xen	pag. 323
2. LXC	pag. 329
1. Le fasi preliminari	pag. 330
2. Configurazione di rete	pag. 330
3. Configurazione del sistema	pag. 331
4. Avvio del container	pag. 332
3. Virtualizzazione con KVM	pag. 334
1. Fasi preliminari	pag. 334
2. Configurazione di rete	pag. 335
3. Installazione in <code>virt-install</code>	pag. 335
4. Gestione delle macchine con <code>virsh</code>	pag. 337
5. Come installare un sistema RPM su Debian attraverso yum	pag. 338
3. Installazione automatizzata	pag. 339
1. Fully Automatic Installer (FAI)	pag. 339
2. Preseeding Debian-Installer	pag. 340
1. Come utilizzare un Preseeding File	pag. 341
2. Creazione di un Preseed File	pag. 341
3. Creazione di un supporto di avvio Media personalizzato	pag. 342
1. Avvio dalla rete	pag. 342
2. Preparazione di una chiave USB di avvio	pag. 342
3. Creare un'immagine CD-Rom	pag. 343
3. Simple-CDD: la soluzione all-in-one	pag. 343
1. Creazione dei profili	pag. 344
2. Come configurare ed impiegare <code>build-simple-cdd</code>	pag. 344
3. Creazione dell'immagine ISO	pag. 345
4. Monitoring [Monitoraggio o Supervisione]	pag. 345
1. Configurazione di Munin	pag. 346
1. Configurazione degli host da monitorare	pag. 346
2. Configurazione del Grapher	pag. 347
2. Configurazione di Nagios	pag. 348
1. Installazione	pag. 348
2. Configurazione	pag. 349

<<Questo capitolo riprende alcuni argomenti già trattati dal punto di vista dei mass-deployment systems e non dell'installazione su un singolo computer; pertanto verrà presa in esame la creazione manuale di una partizione RAID o dei volumi LVM e non quella automatizzata gestita dall'installer, allo scopo di mettervi in condizione di poter ritornare eventualmente sui vostri passi. Dopodiché ci occuperemo degli strumenti di monitoraggio e delle tecniche di virtualizzazione. Per ovvie ragioni questo capitolo potrebbe interessare maggiormente gli amministratori professionisti e non chi si occupa semplicemente della propria rete domestica.>>

12.1. RAID e LVM

Il Capitolo 4, "Installazione" a pagina 48 ha in realtà già introdotto le tecnologie ivi riprese, prendendo però in considerazione solo la loro iniziale integrazione e deployment. Purtroppo questo non basta, in quanto un amministratore dopo un installazione iniziale deve essere in grado di gestire le potenziali esigenze di spazio di archiviazione senza essere costretto ad effettuare nuove installazioni che lo danneggerebbero in termini di costo [non necessariamente di natura economica, ma di tempo]. Di conseguenza gli amministratori dovranno essere anche in condizioni di padroneggiare gli strumenti dedicati alla gestione dei volumi RAID e LVM.

Queste due tecniche consentono di astrarre dalle controparti fisiche (ovvero i dischi rigidi o le loro partizioni) i volumi di cui occorre effettuare il mounting; il RAID è destinato alla protezione dei dati da potenziali guasti hardware per mezzo della ridondanza, mentre la tecnica LVM consente un volume management flessibile ed indipendente rispetto all'effettivo spazio sugli underlying disks [per underlying disks si intendono i dischi coinvolti o sottostanti un metodo, una tecnica, un sistema o un sottosistema]. In entrambi i casi, le due summenzionate tecnologie determinano la creazione di block devices che possono essere destinati ai filesystems oppure al swap space ed il mapping di quest'ultimi non necessariamente coinvolge direttamente un singolo hard disk fisico. Sebbene i metodi RAID ed LVM vengano allestiti in modo differente ed abbiano origini diverse, le loro funzionalità a volte sono sovrapponibili e per tale ragione vengono spesso citati insieme. [Il block device o dispositivo a blocchi o file di dispositivo o device file è un tipo di file che rappresenta una periferica come ad esempio /dev/sda. Lo swap space è lo spazio dedicato allo "swapping" (trad. lett. "scambio") di dati della RAM con una memoria ausiliaria in modo che la RAM possa liberarsi ed occuparsi di altre attività. Il mapping è una tecnica di trasferimento dati che consente la "data accuracy" (trad. lett. "l'accuratezza dei dati") dei dati trasferiti dalla sorgente al duplicato].

IN PROSPETTIVA
Btrfs consente di mettere insieme LVM e RAID

Mentre LVM e RAID sono due distinti kernel subsystems che si frappongono fra i block devices dei dischi ed i loro filesystems, btrfs è un nuovo filesystem, inizialmente sviluppato da Oracle che combina le funzionalità LVM e RAID ed altro ancora. Sebbene sia nella maggior parte dei casi funzionale viene ancora contrassegnato come "sperimentale" in quanto il suo sviluppo è incompleto (ossia alcune sue funzionalità non sono state implementate); è un fatto noto che comunque viene impiegato in diversi production environments. [Production environment è un'espressione tipica del gergo degli sviluppatori che indica l'implementazione negli ambienti degli end users e da parte di quest'ultimi di tecnologie o software].

♦ <http://btrfs.wiki.kernel.org/>

Tra le sue funzionalità più interessanti offre la possibilità di effettuare uno snapshot [trad. lett. "istantanea", in analogia all'istantanea fotografica] dello stato di un filesystem in un dato momento. Inizialmente la copia snapshot non consuma spazio sul disco, in quanto i dati sono in concreto duplicati solo quando viene modificata una delle due copie. Il filesystem si occupa inoltre della transparent compression dei files ed i correlati checksums garantiscono l'integrità di tutti i dati archiviati. [La transparent compression è un metodo attraverso cui viene preservata l'integrità dei dati compressi, in modo che non ci siano differenze fra il duplicato decompresso ed il file sorgente che è stato compresso]

Il kernel supporta entrambe le tecnologie RAID e LVM attraverso block device files, che funzionano similmente ai block device files destinati individualmente ad un disco rigido o ad una partizione. Quando un'applicazione, o un'altra parte del kernel, ha bisogno di accedere ad un blocco corrispondente ad un device, l'appropriate subsystem [trad. lett. il subsystem di competenza] instrada il blocco verso il pertinente physical layer [il livello 1 o livello fisico del modello ISO/OSI che come mezzi trasmissivi impiega mezzi elettrici (doppini telefonici, cavi coassiali, ecc.), mezzi ottici (fibre ottiche, laser, ecc.) e mezzi wireless]. In base alla configurazione, il blocco può essere archiviato su uno o diversi dischi fisici e l'effettiva posizione fisica del blocco potrebbe non avere correlazioni con la posizione del corrispettivo blocco nel logical device.

12.1.1. Software RAID

RAID è l'acronimo di Redundant Array of Independent Disks. Lo scopo di questo sistema è la protezione da perdite dati dovute a guasti del disco rigido. Il principio generale è semplice: i dati vengono salvati su diversi dischi fisici e non su un solo disco, tramite un level of redundancy [trad. lett. "livello di ridondanza"]. In base al livello di ridondanza e qualora si verificasse un guasto improvviso dei dischi, i dati potranno comunque essere ricostruiti senza perdite [loss-lessly in ingl.] attraverso i dischi ancora attivi.

CULTURA Independent o inexpensive?	La I dell'acronimo RAID originariamente era l'iniziale del termine inglese <i>inexpensive</i> (economico), in quanto il sistema RAID in sé consente l'incremento della sicurezza dei dati senza dover investire in dischi high-end. Ma con il tempo, forse per questioni di immagine dato che l'attributo economico poteva essere frainteso come "poco raccomandabile", la I dell'acronimo RAID ha assunto il significato di <i>independent</i> (indipendente).
--	---

Il RAID può essere implementato attraverso hardware dedicato (Integrated RAID modules su controller cards SCSI o SATA) oppure via software abstraction [trad. lett. astrazione software] (ossia tramite il kernel). Che si tratti di hardware o di software, un sistema RAID con sufficiente ridondanza può rimanere operativo (mantenendo la "trasparenza dei dati") anche in caso di guasto di uno dei dischi; inoltre, malgrado il sopraccitato guasto, l'ultimo layer dello stack [con il termine *stack* ci si riferisce alla "pila" ISO/OSI] (applicazioni) può continuare ad accedere ai dati ininterrottamente. Ovviamente tale modalità denominata "degradata" [o degraded mode in ingl.] può avere implicazioni sulle prestazioni, riduce la ridondanza e può infine condurre alla perdita di dati qualora si verificasse un ulteriore guasto di un ulteriore disco. Per praticità il RAID rimane attivo anche in modalità degradata, ma è consigliabile che manteneate la suddetta condizione solo per il tempo necessario per sostituire il disco rotto. E solo quando il nuovo disco subentrerà nel RAID, lo stesso sistema RAID potrà ricostruire i dati necessari per ritornare in modalità sicura (safe mode in ingl.). Ovviamente la suddetta condizione dell'array (struttura dati) viene celata alle applicazioni e quest'ultime potranno avvertirne l'effetto soltanto per la riduzione della velocità durante la modalità degradata o durante la fase di ricostruzione del RAID in sé.

Quando un RAID viene implementato via hardware solitamente se ne potrà ritrovare la configurazione all'interno del setup tool del BIOS ed il kernel Linux considererà il RAID array come se fosse un singolo disco funzionante alla stessa stregua di un disco standard, tranne per il fatto che il device name [trad. lett. nome del dispositivo] potrebbe essere differente. In questo manuale verrà trattato solo il software RAID.

12.1.1.1 I diversi livelli RAID

Il RAID di fatto non è un unico sistema, bensì un insieme di sistemi classificati in livelli; i livelli RAID differiscono fra loro per il layout [disposizione dei dati] e per quantità di ridondanza supportati. Maggiore ridondanza equivale a migliore failure-proof [trad. lett. resistenza ai guasti], in quanto in conseguenza alla quantità di ridondanza il sistema può continuare a funzionare con più dischi guasti. D'altra parte il volume di spazio libero e disponibile si riduce ad una serie di dischi; ovvero sarà necessario disporre di più dischi per conservare la stessa quantità di dati.

Linear RAID Sebbene il RAID subsystem del kernel consenta l'implementazione di un "linear RAID", tale sistema non è strettamente parlando un vero RAID, dato che non viene messa in atto la ridondanza. Il kernel aggrega semplicemente più dischi (in end-to-end) mettendoli a disposizione nelle vesti di un singolo virtual disk (un singolo block device). Si limita solo alla summenzionata funzionalità. Raramente tale configurazione viene impiegata in esclusiva (troverete le eccezioni in seguito), a causa della sua mancanza di ridondanza che comporta l'inaccessibilità del whole aggregate [trad. non lett. dell'intero sistema aggregato] e di conseguenza di tutti i dati qualora si verifichi un guasto di un singolo disco.

RAID-0 Anche questo sistema non offre alcuna ridondanza, ma i dischi non sono semplicemente accatastati uno dopo l'altro: vengono divisi in stripes [attraverso la tecnica denominata data striping, che consiste nella distribuzione] in alternanza [della segmentazione logica dei dati sequenziali] sui diversi dischi fisici. Pertanto, la configurazione RAID-0 con due dischi determinerà la distribuzione dei blocchi del virtual device in base alla seguente logica: i blocchi pari verranno conservati sul primo disco fisico, mentre sul secondo disco fisico verranno archiviati i blocchi dispari.

Dunque l'obiettivo di questo sistema non è incrementare la reliability (affidabilità), dato che come nel linear RAID anche nel RAID-0 la disponibilità dei dati può essere compromessa dal guasto di un singolo disco, ma migliorare le prestazioni:

accedendo in modo sequenziale ad una vasta quantità di dati contigui, il kernel è in condizioni di leggere (o scrivere) in parallelo sui due o più dischi contemporaneamente, di conseguenza incrementando il data transfer rate [o throughput]. Tendenzialmente all'impiego del RAID-0 viene preferita la tecnologia LVM (che verrà trattata più avanti).

RAID-1 Conosciuto anche come "RAID mirroring", è il sistema RAID più semplice e più comunemente utilizzato. La sua configurazione standard prevede l'utilizzo di due dischi fisici della stessa dimensione, da cui viene realizzato un volume logico della stessa dimensione [di un disco]. Il nickname "mirror" (specchio) è dovuto al fatto che i dati vengono archiviati in modo identico sui due dischi. Pertanto se uno dei due dischi si guasta i dati rimangono accessibili sull'altra unità. Per i dati critici potrete impiegare il RAID-1 su più di due dischi anche se ciò inevitabilmente inciderà sul rapporto costi/guadagni fra l'hardware impiegato ed il payload [carico utile o capacità di carico] dello spazio disponibile dei dischi.

NOTA
I dischi ed i
cluster sizes

Se per il RAID mirroring vengono utilizzati due dischi di dimensioni diverse il disco più grande non verrà utilizzato completamente, in quanto dovrà contenere esattamente gli stessi dati di quello più piccolo (e niente di più). Di fatto quindi la capacità disponibile sul volume RAID-1 corrisponde alla dimensione del più piccolo dei dischi che compongono l'array. Questo principio è valido anche per i volumi RAID con un livello più alto, per quanto la ridondanza possa essere distribuita differentemente.

In conclusione dovreste fare attenzione durante la configurazione degli RAID arrays (eccetto per i sistemi RAID-0 e linear RAID) ed assemblare solo dischi della stessa o simile dimensione onde evitare sprechi di risorse. [In realtà per cluster size si intende genericamente anche la più piccola quantità di spazio (unità) sul disco che può essere utilizzata per caricare un file in base al filesystem in uso. Ad esempio con NTFS corrisponde a 4 kilobytes (KB) fino 16 TB di volume. È rilevante in quanto un file deve occupare uno spazio che corrisponda ad un multiplo pari del cluster size. Qualora il multiplo fosse dispari dovrà essere riservato al file dello spazio in più in modo che il multiplo di cluster size sia pari e successivo al multiplo dispari a cui il file effettivamente corrisponde. L'aspetto negativo di tale metodo è che comporta in sostanza la perdita dello spazio disponibile sul disco]

NOTA
Spare disks
[dischi di
salvataggio di
riserva o hot
spare o warm
spare o hot
standby]

Nei livelli di RAID che supportano la ridondanza è possibile aggiungere dei dischi in più rispetto al requisito minimo per costituire l'array. I dischi extra potranno essere impiegati per fungere da spares [dischi di salvataggio di riserva] qualora uno dei dischi principali dell'array dovesse guastarsi. Per esempio su un array di due dischi è possibile aggiungere un disco da spare in modo che il kernel possa ricostruire automaticamente il mirror dallo spare disk, garantendo nuovamente la ridondanza dopo la fase di ricostruzione. Questo sistema è un ulteriore difesa per i dati veramente critici. Vi potreste chiedere quali benefici ci sono ad utilizzare il summenzionato sistema rispetto ad un RAID mirroring su tre dischi. Il principale vantaggio è che uno spare disk può essere condiviso tra più volumi RAID. Ad esempio con tre volumi duplicati attraverso mirroring potrete garantire la ridondanza in caso di guasto dei dischi principali con soli 7 dischi (tre copie, più uno spare disk condiviso) invece di 9 dischi (suddivisi in tre triplette).

Questo livello RAID, nonostante possa comportare dei costi (dato che solo la metà del suo spazio fisico può essere di fatto utilizzabile), è comunemente impiegato. Il RAID-1 è semplice da comprendere ed agevole per i backups: dato che è composto da due dischi identici, uno dei due può essere estratto temporaneamente senza incidere sul funzionamento del working system. Le prestazioni di lettura del RAID-1 sono generalmente migliori rispetto a quelle di un singolo disco in quanto il sistema può teoricamente leggere metà dei dati su ciascun disco in parallelo, senza che la degradazione della velocità di scrittura sia troppo considerevole. Il RAID-1 array con N dischi garantisce la disponibilità dei dati anche in caso di guasti N-1.

RAID-4 Questo sistema RAID, che non è comunemente impiegato, utilizza N dischi per memorizzare i dati utili ed un disco aggiuntivo per la redundancy information. Se quest'ultimo disco smette di funzionare, il sistema può ricostruirla utilizzando gli altri N dischi. Se uno degli N dischi utilizzati per i dati si guasta, i restanti N-1 dischi insieme con il disco di parità, dato che contengono informazioni sufficienti, potranno essere impiegati per ricostruire i dati necessari.

Il sistema RAID-4 non è dispendioso dato che comporta soltanto un aumento di 1 su N dischi in termini di costi, non ha un impatto notevole sulle prestazioni di lettura ma solo sulla scrittura. Tuttavia l'aspettativa di vita (o durata - life expectancy - lifespan) del disco di parità viene ridotta drasticamente dal fatto che la scrittura su uno degli N dischi ne determina la sua scrittura con un numero conseguente di scritture maggiore rispetto agli N dischi. I dati su un RAID-4 array saranno preservati solo se si guasta fino ad 1 disco (su N+1).

RAID-5 Il RAID-5 ha risolto l'asimmetria del RAID-4; i blocchi di parità sono distribuiti sugli N + 1 dischi e così facendo nessun disco è dedicato ad un unico scopo.

Le prestazioni di lettura e scrittura sono pressoché invariate rispetto ad un RAID-4. Anche in questo caso, il RAID-5 è in grado di rimanere funzionante solo se si guasta non di più di un disco degli N+1 dischi.

RAID-6 Il RAID-6 in sostanza può essere considerato come un RAID-5 più esteso, in cui ad ogni serie di blocchi sugli N dischi vengono associati blocchi di parità distribuiti sugli N + 2 dischi.

Questo livello RAID è leggermente più costoso dei due precedenti, ma offre una protezione dei dati più efficiente dato che è in grado di preservare tutti i dati anche in caso di guasto simultaneo di due dischi (su N+2). D'altra parte il processo di scrittura di questo sistema coinvolge sia un blocco dati, sia due blocchi di ridondanza, determinando un'ulteriore degenerazione delle prestazioni di scrittura.

RAID-1+0 Non si tratta in senso stretto di un livello RAID, bensì di una pila RAID composta da due livelli RAID [denominati RAID annidati]. Partendo da $2 \times N$ dischi, occorre appaiarli in N volumi RAID-1. Questi N volumi possono essere aggregati in un unico volume attraverso un RAID lineare o (di frequente) attraverso un LVM. In quest'ultimo caso, si va ben oltre un RAID puro, ma non nuoce. Il RAID-1+0 può continuare a funzionare anche se si guastano più dischi: fino a N dell'array $2 \times N$ descritto sopra, a condizione che almeno un disco di ogni coppia RAID-1 rimanga funzionante.

ANDANDO OLTRE RAID-10	Per "RAID-10" generalmente si intende un "RAID-1+0", generalizzazione di fatto dovuta alla sua specificità per Linux. La configurazione "RAID-10" consente la realizzazione di un sistema in cui ogni blocco viene duplicato su due dischi diversi, persino se il numero di dischi è dispari, e le copie sono distribuite in base al modello configurato. Le prestazioni possono variare in base alla scelta del modello per la ripartizione e del livello di ridondanza, nonché del workload [trad. non lett. carico di attività] del volume logico.
----------------------------------	--

Ovviamente dovrete scegliere il livello di RAID in base ai vostri vincoli ed alle specifiche necessarie per ciascuna applicazione. Inoltre si precisa che un singolo computer è compatibile con diversi tipi di RAID arrays e con differenti configurazioni.

12.1.1.2 Come configurare un RAID

I volumi RAID vengono configurati attraverso il pacchetto mdadm; questo pacchetto contiene l'omonimo comando mdadm, che consente la creazione e la gestione dei RAID arrays, per mezzo di scripts e tools per l'integrazione con il sistema, tra cui un sistema di monitoraggio. Prendiamo ad esempio un server con un numero di dischi collegati, di cui solo alcuni sono disponibili per configurare il RAID in quanto gli altri sono già impiegati in altre attività. Inizialmente pertanto avremo i seguenti dischi e le seguenti partizioni:

- il disco sdb, 4 GB, è completamente disponibile;
- il disco sdc, 4 GB, è completamente disponibile;
- sul disco sdd è disponibile solo la partizione sdd2 di circa 4 GB;
- infine il disco sde, 4 GB, è completamente disponibile.

NOTA
Identificare i volumi RAID già esistenti

Il file /proc/mdstat elenca i volumi già esistenti ed il loro stato. Pertanto quando creerete un nuovo volume RAID, prestate attenzione a non denominarlo con il nome di un volume già esistente.

Tramite i suddetti physical elements verranno realizzati due volumi, un RAID-0 ed un mirror (RAID-1). Dapprima occorre procedere con la creazione del RAID-0:

```
# mdadm --create /dev/md0 --level=0 --raid-devices=2 /dev/sdb /dev/sdc
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.

# mdadm --query /dev/md0
/dev/md0: 8.00GiB raid0 2 devices, 0 spares. Use mdadm --detail for more detail.

# mdadm --detail /dev/md0
/dev/md0:
      Version : 1.2
      Creation Time : Wed May  6 09:24:34 2015
      Raid Level : raid0
      Array Size : 8387584 (8.00 GiB 8.59 GB)
      Raid Devices : 2
      Total Devices : 2
      Persistence : Superblock is persistent

      Update Time : Wed May  6 09:24:34 2015
      State : clean
      Active Devices : 2
      Working Devices : 2
      Failed Devices : 0
      Spare Devices : 0

      Chunk Size : 512K

            Name : mirwiz:0  (local to host mirwiz)
            UUID : bb085b35:28e821bd:20d697c9:650152bb
            Events : 0

Number Major Minor RaidDevice State
 0        8      16      0      active sync /dev/sdb
 1        8      32      1      active sync /dev/sdc

# mkfs.ext4 /dev/md0
mke2fs 1.42.12 (29-Aug-2014)
Creating filesystem with 2095104 4k blocks and 524288 inodes
Filesystem UUID: fff08295-bebe-41a9-9c6a-8c7580e520a6
Superblock backups stored on blocks:
```

```
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632
```

```
Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
# mkdir /srv/raid-0
# mount /dev/md0 /srv/raid-0
# df -h /srv/raid-0
Filesystem Size Used Avail Use% Mounted on
/dev/md0 7.9G 18M 7.4G 1% /srv/raid-0
```

Il comando mdadm --create richiede diversi parametri: il nome del volume da creare (/dev/md*, MD è l'acronimo di Multiple Device), il livello RAID, il numero di dischi (tale parametro è obbligatorio nonostante abbia senso soltanto a partire dal RAID-1 in su) ed i physical drives da utilizzare. Una volta creato il device, potrete utilizzarlo come una normale partizione, creare un filesystem, eseguire il mounting del filesystem, ecc. Occorre precisare che la numerazione dell'array del volume RAID-0 appena creato su md0 è soltanto una coincidenza e non ha alcun significato particolare o correlazione con la ridondanza scelta. Volendo potrete creare dei RAID arrays denominati in altro modo, attraverso i parametri del comando mdadm ed assegnare /dev/md/linear invece di /dev/md0.

La creazione di un volume RAID-1 viene implementata in modo simile e le differenze sono percettibili solo dopo la sua creazione:

```
# mdadm --create /dev/md1 --level=1 --raid-devices=2 /dev/sdd2 /dev/sde
mdadm: Note: this array has metadata at the start and
may not be suitable as a boot device. If you plan to
store '/boot' on this device please ensure that
your boot-loader understands md/v1.x metadata, or use
--metadata=0.90
mdadm: largest drive (/dev/sdd2) exceeds size (4192192K) by more than 1%
Continue creating array? y
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md1 started.
# mdadm --query /dev/md1
/dev/md1: 4.00GiB raid1 2 devices, 0 spares. Use mdadm --detail for more detail.
# mdadm --detail /dev/md1
/dev/md1:
      Version : 1.2
      Creation Time : Wed May  6 09:30:19 2015
      Raid Level : raid1
      Array Size : 4192192 (4.00 GiB 4.29 GB)
      Used Dev Size : 4192192 (4.00 GiB 4.29 GB)
      Raid Devices : 2
      Total Devices : 2
      Persistence : Superblock is persistent

      Update Time : Wed May  6 09:30:40 2015
      State : clean, resyncing (PENDING)
```

```

Active Devices : 2
Working Devices : 2
Failed Devices : 0
Spare Devices : 0

        Name : mirwiz:1  (local to host mirwiz)
        UUID : 6ec558ca:0c2c04a0:19bca283:95f67464
Events : 0

```

Number	Major	Minor	RaidDevice	State
0	8	50	0	active sync /dev/sdd2
1	8	64	1	active sync /dev/sde

```

# mdadm --detail /dev/md1
/dev/md1:
[...]
State : clean
[...]

```

SUGGERIMENTO
RAID, dischi e partizioni

Dall'esempio soprastante è evidente che i RAID devices possono essere realizzati su partizioni e non necessariamente su interi dischi.

Arrivati a questo punto occorre fare delle precisazioni. Innanzitutto mdadm ha rilevato che i due elementi fisici non hanno la stessa dimensione; dato che verrà sprecato lo spazio in più dell'elemento più grande fra i due, viene richiesta la conferma per proseguire.

Dopodiché occorre verificare lo status del mirror. Lo status "standard" del mirror presuppone infatti che i due dischi abbiano un contenuto pressoché identico. Ma la prima creazione del volume non offre garanzie. Pertanto se ne occuperà il sottosistema del RAID attraverso una fase di sincronizzazione (synchronization phase) che si avvierà alla creazione del dispositivo RAID. Dopo qualche istante (a seconda delle dimensioni dei dischi...), il RAID array dichiarerà lo switch allo status "active" (attivo) o "clean" (pulito). Va precisato che durante la fase di ricostruzione il mirror si trova in modalità degradata e la ridondanza non è garantita. Di conseguenza il guasto di uno dei dischi durante la suddetta risk window può comportare la perdita di tutti i dati. [risk window è un termine solitamente utilizzato in medicina e viene comunemente tradotto come periodo finestra o finestra di rischio. Alla stessa stregua del paradosso del gatto di Schrödinger, indica che durante un dato lasso di tempo non è possibile determinare un esito positivo o negativo]. In ogni caso è raro che un RAID array appena creato contenga una grande mole di dati critici prima della initial synchronization [trad. lett. sincronizzazione iniziale]. Si precisa che durante la modalità degradata, il dispositivo /dev/md1 è comunque disponibile e quindi potrete al suo interno creare un filesystem e copiare i dati.

SUGGERIMENTO
Come avviare un mirror in modalità degradata

Poniamo il caso che desideriate realizzare un RAID-1 mirror e non sono direttamente disponibili due dischi, in quanto uno dei due dischi necessari per realizzare l'array è già in uso e contiene i dati da migrare nello stesso array. Per rimediare potrete creare deliberatamente un RAID-1 array in modalità degradata, utilizzando come argomento per il comando mdadm il termine missing invece di un device file. Una volta che i dati sono stati copiati nel "mirror", il disco originale può essere integrato nell'array. Avrà luogo quindi una sincronizzazione e dopodiché la ridondanza dei dati desiderata.

SUGGERIMENTO
Come configurare
un mirror
privo di
sincronizzazione

I volumi RAID-1 sono solitamente creati per essere utilizzati come un disco nuovo, un disco vergine. Pertanto il contenuto originale di quel disco non conta; diversamente i dati scritti dopo la creazione del volume e la loro accessibilità hanno importanza, in particolare il filesystem. Potreste chiedervi a questo punto se occorre sincronizzare i due dischi del mirror durante la sua creazione. Difatti a quale scopo rendere identiche due zone del volume che verranno “lette” solo quando conterranno effettivamente dei dati scritti? Fortunatamente, la fase di sincronizzazione può essere evitata per mezzo dell’opzione `--assume-clean` di mdadm. Tuttavia questa opzione non è abilitata di default in quanto può comportare degli effetti indesiderati nei casi d’uso in cui occorre poter leggere gli initial data [trad. lett. “i dati originali”] (in particolare se è già presente un filesystem).

Occorre prendere in considerazione un eventuale guasto di uno degli elementi del RAID-1 array. mdadm consente di simulare il guasto per mezzo dell’opzione `--fail`

```
# mdadm /dev/md1 --fail /dev/sde
mdadm: set /dev/sde faulty in /dev/md1
# mdadm --detail /dev/md1
/dev/md1:
[...]
      Update Time : Wed May  6 09:39:39 2015
      State : clean, degraded
  Active Devices : 1
Working Devices : 1
 Failed Devices : 1
   Spare Devices : 0
          Name : mirwiz:1  (local to host mirwiz)
          UUID : 6ec558ca:0c2c04a0:19bca283:95f67464
          Events : 19
Number Major Minor RaidDevice State
    0      8        50      0      active sync  /dev/sdd2
    2      0        0       2      removed
    1      8        64      -      faulty   /dev/sde
```

Nonostante il gusto i contenuti del volume sono ancora accessibili (inoltre alle applicazioni non comporta alcuna conseguenza se il disco guasto è ancora “mounted”), ma la sicurezza dei dati non è più garantita: pertanto se il disco sdd dovesse guastarsi, i dati andrebbero persi per sempre. Onde evitare tale rischio, è necessario procedere con la sostituzione del disco guasto con uno nuovo, sdf:

```

# mdadm /dev/md1 --add /dev/sdf
mdadm: added /dev/sdf
# mdadm --detail /dev/md1
/dev/md1:
[...]
    Raid Devices : 2
    Total Devices : 3
        Persistence : Superblock is persistent

        Update Time : Wed May  6 09:48:49 2015
                    State : clean, degraded, recovering
    Active Devices : 1
Working Devices : 2
 Failed Devices : 1
  Spare Devices : 1

Rebuild Status : 28% complete

        Name : mirwiz:1 (local to host mirwiz)
        UUID : 6ec558ca:0c2c04a0:19bca283:95f67464
        Events : 26

      Number  Major  Minor  RaidDevice  State
          0      8      50        0  active sync /dev/sdd2
          2      8      80        1  spare  rebuilding /dev/sdf

          1      8      64        -  faulty /dev/sde
# [...]
[...]
# mdadm --detail /dev/md1
/dev/md1:
[...]
    Update Time : Wed May  6 09:49:08 2015
                    State : clean
    Active Devices : 2
Working Devices : 2
 Failed Devices : 1
  Spare Devices : 0

        Name : mirwiz:1 (local to host mirwiz)
        UUID : 6ec558ca:0c2c04a0:19bca283:95f67464
        Events : 41

      Number  Major  Minor  RaidDevice  State
          0      8      50        0  active sync /dev/sdd2
          2      8      80        1  active sync /dev/sdf

          1      8      64        -  faulty /dev/sde

```

Anche in questo caso il kernel metterà in atto automaticamente una fase di ricostruzione, durante la quale il volume, seppur accessibile, sarà in modalità degradata. Una volta completata la fase di ricostruzione, il RAID array tornerà al suo status normale. A questo punto potrete comunicare al sistema di rimuovere il disco sde dall'array e vi ritroverete con un classico RAID mirror con due dischi:

```
# mdadm /dev/md1 --remove /dev/sde
mdadm: hot removed /dev/sde from /dev/md1
# mdadm --detail /dev/md1
/dev/md1:
[...]
Number      Major      Minor    RaidDevice   State
          0          8          50          0  active sync  /dev/sdd2
          2          8          80          1  active sync  /dev/sdf
```

Il disco potrà quindi essere rimosso fisicamente allo spegnimento del server oppure mentre è ancora accesso se si tratta di un'unità host-removed ossia un'unità hot swap che può essere rimossa a caldo. Le summenzionate configurazioni sono compatibili con la maggior parte degli SCSI controllers, SATA disks ed unità esterne su USB o Firewire.

12.1.1.3 Backup della configurazione

La maggior parte dei metadati dei volumi RAID viene salvata direttamente sui dischi che compongono lo stesso array, in modo che il kernel possa rilevare gli arrays, le componenti e assemblare tutto automaticamente all'avvio del sistema. Tuttavia, è consigliabile effettuare un backup della configurazione dato che il rilevamento non è "a prova di guasto" e si presenterà nei momenti meno opportuni. Se il guasto del disco sde dell'esempio soprastante fosse stato reale e se il sistema fosse stato riavviato senza aver rimosso il disco guasto, quest'ultimo sarebbe tornato in funzione nonostante tutto, in quanto rilevato durante il reboot. Il kernel quindi si ritroverebbe con tre elementi fisici che dichiarano contemporaneamente di contenere la metà dello stesso volume RAID. Un'ulteriore sorgente di confusione potrebbe presentarsi se i volumi RAID provengono da due server distinti ed assemblati poi sotto uno dei due server. Se gli arrays funzionavano correttamente prima di essere trasferiti, il kernel sarà in grado di riassemblare correttamente le "coppie"; ma se i dischi del vecchio server sono stati aggregati come md1 ed un mirror md1 sul nuovo server esiste già, potrebbe essere necessario cambiargli nome.

Ecco la necessità di un backup della configurazione anche se per mera consultazione. Il metodo standard prevede la modifica del file /etc/mdadm/mdadm.conf, di cui a seguire troverete un esempio.

Esempio 12.1 File di configurazione mdadm

```
# mdadm.conf
#
# Please refer to mdadm.conf(5) for information about this file.
#
# by default (built-in), scan all partitions (/proc/partitions) and all
# containers for MD superblocks. alternatively, specify devices to scan, using
# wildcards if desired.
```

```

DEVICE /dev/sd*

# auto-create devices with Debian standard permissions
CREATE owner=root group=disk mode=0660 auto=yes

# automatically tag new arrays as belonging to the local system
HOMEHOST <system>

# instruct the monitoring daemon where to send mail alerts
MAILADDR root

# definitions of existing MD arrays
ARRAY /dev/md0 metadata=1.2 name=mirwiz:0
UUID=bb085b35:28e821bd:20d697c9:650152bb
ARRAY /dev/md1 metadata=1.2 name=mirwiz:1
UUID=6ec558ca:0c2c04a0:19bca283:95f67464

# This configuration was auto-generated on Thu, 17 Jan 2013 16:21:01 +0100
# by mkconf 3.2.5-3

```

Una delle informazioni più utili è l'opzione DEVICE, che specifica l'elenco dei dispositivi su cui il sistema cercherà di rilevare automaticamente le componenti dei volumi RAID allo start-up time. Nell'esempio sono stati sostituiti il default value, i partitions containers con un elenco esplicito di device files dal momento che si è scelto di utilizzare dei dischi interi invece delle partizioni per i diversi volumi.

Le ultime due righe dell'esempio soprastante sono quelle che consentono al kernel di scegliere in sicurezza quale volume number [o volume serial number da non confondere con il volume label] da associare ad un dato array. I metadati conservati negli stessi dischi sono infatti sufficienti per riassemblare i volumi, ma non per determinarne il volume number (e di conseguenza il device name associato /dev/md*).

Fortunatamente, queste righe possono essere generate automaticamente:

```

# mdadm --misc --detail --brief /dev/md?
ARRAY /dev/md0 metadata=1.2 name=mirwiz:0
UUID=bb085b35:28e821bd:20d697c9:650152bb
ARRAY /dev/md1 metadata=1.2 name=mirwiz:1
UUID=6ec558ca:0c2c04a0:19bca283:95f67464

```

Il contenuto di queste ultime due righe non dipende dall'elenco dei dischi che compongono i volumi. Potrete quindi fare a meno di rigenerarle per la sostituzione di un disco difettoso con una nuovo. Tuttavia, dovete aggiornarle dopo ogni creazione o eliminazione di RAID array.

12.1.2. LVM

Il sistema LVM, o Logical Volume Manager, è un altro approccio per l'astrazione dei volumi logici dai loro stessi supporti fisici ed il suo obiettivo principale è di migliorare la flessibilità e non l'affidabilità. Difatti LVM permette di modificare dinamicamente un volume logico, in modo trasparente per le applicazioni [ovvero quest'ultime non dovranno essere modificate]; pertanto potrete aggiungere nuovi dischi, migrare i dati su di loro e rimuovere i vecchi dischi, senza effettuare l'unmounting del volume.

12.1.2.1 I Concetti LVM

La summenzionata flessibilità viene raggiunta attraverso un livello di astrazione che racchiude tre concetti. [L'impiego del termine concetto nell'applicazione del metodo logico di astrazione in informatica è dovuto al suo significato in senso lato di idea astratta e generale]

Innanzitutto, il PV (Physical Volume) o volume fisico in italiano è l'entità per eccellenza riservata all'hardware: può essere una partizione su disco o un disco intero o un dispositivo a blocchi (come ad esempio un RAID array). Fate attenzione però, se configurerete un elemento fisico come PV per un LVM, potrete accedervi solo attraverso lo stesso sistema LVM, altrimenti rischierete di sconvolgere il sistema LVM.

Diversi PV possono essere raggruppati in cluster al di sotto di un VG (Volume Group) [in ital. gruppo di volumi, gruppo volume] e possono essere considerati alla stessa stregua dei dischi virtuali ed espandibili. I VG sono astratti e non sono elencati come device file in /dev/, quindi non c'è il rischio di di potervi accedere direttamente.

Infine il terzo tipo di oggetto è il LV (Logical Volume) [volume logico in ital.], che è un chunk di un VG; riprendendo il paragone del VG con i dischi, un LV può essere comparato ad una partizione. Il volume logico viene elencato come block device in /dev e può essere utilizzato come qualsiasi partizione fisica (solitamente viene utilizzato per ospitare un filesystem o lo swap space). [uno swap space è uno spazio sul disco dedicato allo swapping; un chunck di un VG, trad. lett. porzione di un VG, è di fatto un insieme di informazioni che consentono ad un LV di essere elencato e disponibile fra i block devices]

Si precisa inoltre che la suddivisione di un VG in volumi LV non è assolutamente correlata alle componenti fisiche PV del VG. Di conseguenza un VG in un singolo componente fisico (ad esempio un disco) può essere suddiviso in una dozzina di volumi logici; allo stesso modo un VG può essere distribuito su più dischi fisici ed apparire come un singolo vasto volume logico. L'unico limite, ovviamente, è che il total size [la grandezza totale del blocco] destinato al LV non deve superare la capacità totale dei PV nel gruppo di volumi.

Ciò non toglie che nel raggruppare un VG è consigliabile applicare una certa omogeneità nella selezione delle componenti fisiche, nonché mettere in atto una suddivisione in volumi logici in base a dei modelli d'uso noti [usage patterns]. Di conseguenza, se disponete sia di dischi veloci, sia di dischi lenti, è possibile raggruppare quelle veloci in un VG, mentre quelle lente in un altro; i chunks del primo Volume Group possono essere assegnate ad attività che richiedono velocità di accesso dati elevate, diversamente i chunks del secondo VG possono essere riservate ad attività di cui si usufruisce meno.

In ogni caso, dovrete tenere presente anche che un LV non ha un stretto rapporto di dipendenza ad un PV. Potrete stabilire la posizione fisica in cui vengono scritti i dati LV, ma tale funzionalità per attività quotidiane è superflua. Al contrario: se il set delle componenti fisiche di un VG viene modificato, le posizioni fisiche nello storage corrispondenti a dei determinati LV potranno essere migrate fra i dischi (rimanendo ovviamente conservate pure all'interno dei volumi fisici che compongono il gruppo di volumi).

12.1.2.2 Come configurare un sistema LVM

In questo paragrafo troverete le fasi della procedura di configurazione di un sistema LVM per un tipico caso d'uso: un contesto storage complesso [dispositivi hardware, supporti per la memorizzazione, infrastrutture e software dedicati, ecc.]. Spesso situazioni simili sono il risultato di soluzioni temporanee che si sono accumulate nel tempo. Ad esempio consideriamo che la domanda di storage da includere in un dato server sia cambiata nel corso del tempo con l'effetto di una configurazione complessa di partizioni disponibili, suddivise su diversi dischi. Per essere più concreti si considerino le potenziali partizioni per il sopra citato esempio:

- sul disco sdb, una partizione sdb2 da 4 GB;
- sul disco sdc, una partizione sdc3 da 3 GB;

- il disco sdd da 4 GB è completamente disponibile;
- sul disco sdf, una partizione sdf1 da 4 GB ed una sdf2 da 5 GB.

Inoltre si presupponga che i dischi sdb e sdf abbiano prestazioni migliori rispetto agli altri due. L'obiettivo è configurare tre volumi logici, per tre distinte applicazioni: un file server (che necessita di 5 GB di spazio di archiviazione), un database (1 GB) ed anche spazio per i back-ups (12 GB). Le prime due attività hanno l'esigenza di prestazioni non indifferenti, ma i back-ups sono meno pretenziosi (o critici) in termini di access speed (trad. lett. velocità di accesso). I summenzionati vincoli impediscono che vengano usate le partizioni singolarmente; occorre pertanto utilizzare un LVM in quanto può effettuare l'astrazione delle dimensioni fisiche dei singoli devices [superandone i limiti fisici] in modo che la configurazione sia "trattenuta" soltanto dalla capacità totale dell'intero storage.

I tools richiesti sono il pacchetto lvm2 e le correlate dipendenze. Se installati, la configurazione di un sistema LVM avviene in tre fasi, che corrispondono ai tre livelli dei tre concetti di un sistema LVM.

Innanzitutto occorre preparare i volumi fisici utilizzando pvcreate:

```
# pvdisplay
# pvcreate /dev/sdb2
Physical volume "/dev/sdb2" successfully created
# pvdisplay
"/dev/sdb2" is a new physical volume of "4.00 GiB"
--- NEW Physical volume ---
PV Name /dev/sdb2
VG Name
PV Size 4.00 GiB
Allocatable NO
PE Size 0
Total PE 0
Free PE 0
Allocated PE 0
PV UUID 0zuiQQ-j1Oe-P593-4tsN-9FGy-TY0d-Quz31I

# for i in sdc3 sdd sdf1 sdf2 ; do pvcreate /dev/$i ; done
Physical volume "/dev/sdc3" successfully created
Physical volume "/dev/sdd" successfully created
Physical volume "/dev/sdf1" successfully created
Physical volume "/dev/sdf2" successfully created

# pvdisplay -C
PV VG Fmt Attr PSize PFree
/dev/sdb2 lvm2 --- 4.00g 4.00g
/dev/sdc3 lvm2 --- 3.09g 3.09g
/dev/sdd lvm2 --- 4.00g 4.00g
/dev/sdf1 lvm2 --- 4.10g 4.10g
/dev/sdf2 lvm2 --- 5.22g 5.22g
```

Niente di complicato insomma; si precisa che un PV può essere configurato sia su un intero disco, sia su partizioni individuali di un disco. Come mostrato nel soprastante esempio, il comando pvdisplay è in grado di elencare i PVs esistenti in due formati output.

Giunti a questo punto occorre assemblare gli elementi fisici nei gruppi di volumi (VG) utilizzando il comando `vgcreate`. Per ovvie ragioni soltanto i PVs dei dischi veloci verranno collocati in `vg_critical` VG mentre i PVs appartenenti ai dischi lenti verranno inclusi in `vg_normal` VG.

```
# vgdisplay
No volume groups found
# vgcreate vg_critical /dev/sdb2 /dev/sdf1
Volume group "vg_critical" successfully created
# vgdisplay
--- Volume group ---
VG Name vg_critical
System ID
Format lvm2
Metadata Areas 2
Metadata Sequence No 1
VG Access read/write
VG Status resizable
MAX LV 0
Cur LV 0
Open LV 0
Max PV 0
Cur PV 2
Act PV 2
VG Size 8.09 GiB
PE Size 4.00 MiB
Total PE 2071
Alloc PE 0 / 0
Free PE 2071/8.09 GiB
VG UUID bpq7zO-PzPD-R7HW-V8eN-c10c-S32h-f6rKqp

# vgcreate vg_normal /dev/sdc3 /dev/sdd /dev/sdf2
Volume group "vg_normal" successfully created
# vgdisplay -C
VG      #PV #LV #SN Attr   VSize   VFree
vg_critical  2    0    0 wz--n-  8.09g   8.09g
vg_normal    3    0    0 wz--n- 12.30g  12.30g
```

Anche in questo caso, i comandi sono piuttosto inequivocabili (e `vgdisplay` dispone di due formati di output). Si puntualizza che niente vi impedisce di utilizzare due partizioni dello stesso disco fisico in due differenti VGs; anche il prefisso `vg_` non è obbligatorio e viene utilizzato nell'esempio soprastante per convenzione.

I due "dischi virtuali" creati corrispondono rispettivamente ad un total size di circa 8 GB e 12 GB, che possono essere suddivisi in "partizioni virtuali" (LV). Ciò viene fatto tramite il comando `lvcreate` e la sua sintassi è un po' più complessa:

```

# lvdisplay
# lvcreate -n lv_files -L 5G vg_critical
Logical volume "lv_files" created
# lvdisplay
--- Logical volume ---
LV Path /dev/vg_critical/lv_files
LV Name lv_files
VG Name vg_critical
LV UUID J3V0oE-cBYO-KyDe-5e0m-3f70-nv0S-kCWbpT
LV Write Access read/write
LV Creation host, time mirwiz, 2015-06-10 06:10:50 -0400
LV Status available
# open 0
LV Size 5.00 GiB
Current LE 1280
Segments 2
Allocation inherit
Read ahead sectors auto
- currently set to 256
Block device 253:0

# lvcreate -n lv_base -L 1G vg_critical
Logical volume "lv_base" created
# lvcreate -n lv_backups -L 12G vg_normal
Logical volume "lv_backups" created
# lvdisplay -C
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync
-> Convert
lv_base vg_critical -wi-a--- 1.00g
lv_files vg_critical -wi-a--- 5.00g
lv_backups vg_normal -wi-a--- 12.00g

```

Per creare i volumi logici sono fondamentali due informazioni; tali informazioni dovranno essere trasmesse tramite le opzioni di `lvcreate`. Il nome di ciascun LV da creare viene specificato attraverso l'opzione `-n`, mentre la sua dimensione viene descritta da `-L`. Ovviamente bisogna anche indicare il VG in cui si vuole creare il LV, ossia l'ultimo parametro di `lvcreate` da riga di comando nell'esempio soprastante.

ANDANDO OLTRE Le opzioni di <code>lvcreate</code>	Il comando <code>lvcreate</code> dispone di diverse opzioni che consentono il <i>tweaking</i> per la creazione di un LV. [Per <i>tweaking</i> si intende la lavorazione o la modifica di un sistema complesso attraverso un metodo fine-tuning (trad. lett. a dosaggio "raffinato").] L'opzione <code>-l</code> consente di specificare il total size (la capacità totale dei blocchi) di un LV in un'unità di misura riferita al numero di blocchi in sé e non in un'unità di misura più intuitiva per gli esseri umani come negli esempi soprastanti. I suddetti blocchi (denominati PE, physical extents, [trad. lett. estensioni fisiche]) sono unità di spazio di archiviazione contigue, che fanno parte dei PVs e non possono essere distribuite tra i LVs. Pertanto se desiderate definire con precisione lo spazio di archiviazione assegnato ad un LV, per usufruire ad esempio dell'intero spazio di archiviazione disponibile, potrete usare l'opzione <code>-l</code> piuttosto che <code>-L</code> . Potrete anche specificare la posizione fisica di un LV, in modo che le sue estensioni fisiche vengano memorizzate in un dato PV (ovviamente un PV del VG predefinito).
--	--

Qualora desideriate privilegiare un database server rispetto ad un file server e premesso che sdb sia più veloce di sdf, potrete caricate lv_base su sdb. Da riga di comando scriverete: lvcreate -n lv_base -L 1G vg_critical /dev/sdb2. Si precisa che il suddetto comando potrebbe non essere in grado di eseguire quanto richiesto se il PV non ha sufficienti estensioni fisiche libere. Riprendendo l'esempio soprastante, per evitare qualsiasi inconveniente, dovreste creare lv_base prima di lv_files oppure liberare spazio su sdb2 attraverso il comando pvmove.

I volumi logici, una volta creati, vengono rappresentati come block device files ed elencati in /dev/mapper/:

```
# ls -l /dev/mapper
total 0
crw----- 1 root root 10, 236 Jun 10 16:52 control
lrwxrwxrwx 1 root root 7 Jun 10 17:05 vg_critical-lv_base -> ../dm-1
lrwxrwxrwx 1 root root 7 Jun 10 17:05 vg_critical-lv_files -> ../dm-0
lrwxrwxrwx 1 root root 7 Jun 10 17:05 vg_normal-lv_backups -> ../dm-2
# ls -l /dev/dm-
brw-rw---T 1 root disk 253, 0 Jun 10 17:05 /dev/dm-0
brw-rw---- 1 root disk 253, 1 Jun 10 17:05 /dev/dm-1
brw-rw---- 1 root disk 253, 2 Jun 10 17:05 /dev/dm-2
```

NOTA
Rilevamento
automatico dei
volumi LVM

All'avvio del computer, l'unit service [un file che contiene la configurazione di un dato servizio] di systemd denominato lvm2-activation mette in esecuzione vgchange -aay per attivare i gruppi di volumi: vengono rilevati i devices disponibili; i devices inizializzati per essere dei volumi fisici per la tecnologia LVM vengono registrati nel sottosistema LVM; i devices che fanno parte dei gruppi di volumi vengono assemblati ed i volumi logici pertinenti vengono attivati e resi disponibili. Pertanto, non è necessario modificare i files di configurazione durante il processo di creazione dei volumi LVM o a seguito di una loro modifica. Si precisa tuttavia che la struttura dei vari elementi inclusi nel sistema LVM è impressa in /etc/lvm/backup e può essere utilizzata in caso di anomalie (o per effettuare delle revisioni).

Per maggiore comodità vengono anche creati automaticamente dei collegamenti simbolici nelle directory corrispondenti ai VGs:

```
# ls -l /dev/vg_critical
total 0
lrwxrwxrwx 1 root root 7 Jun 10 17:05 lv_base -> ../dm-1
lrwxrwxrwx 1 root root 7 Jun 10 17:05 lv_files -> ../dm-0
# ls -l /dev/vg_normal
total 0
lrwxrwxrwx 1 root root 7 Jun 10 17:05 lv_backups -> ../dm-2
```

Di conseguenza potrete utilizzare i LVs proprio come se fossero delle partizioni standard:

```
# mkfs.ext4 /dev/vg_normal/lv_backups
mke2fs 1.42.12 (29-Aug-2014)
Creating filesystem with 3145728 4k blocks and 786432 inodes
Filesystem UUID: b5236976-e0e2-462e-81f5-0ae835ddab1d
[...]
```

```

Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
# mkdir /srv/backups
# mount /dev/vg_normal/lv_backups /srv/backups
# df -h /srv/backups
Filesystem                      Size  Used   Avail   Use%  Mounted on
/dev/mapper/vg_normal-lv_backups    12G   30M    12G    1%   /srv/backups
# [...]
[...]
# cat /etc/fstab
[...]
/dev/vg_critical/lv_base          /srv/base     ext4 defaults 0 2
/dev/vg_critical/lv_files         /srv/files    ext4 defaults 0 2
/dev/vg_normal/lv_backups        /srv/backups  ext4 defaults 0 2

```

Dal punto di vista delle applicazioni finalmente le numerose piccole partizioni sono state astratte in un vasto volume di 12 GB, con un nome associato decisamente più intuitivo.

12.1.2.3 La tecnologia LVM ed il manifestarsi di nuove esigenze nel corso del tempo

Per quanto la capacità di aggregare partizioni o dischi fisici possa essere lodevole, non rappresenta l'asso nella manica della tecnologia LVM. Difatti la sua peculiare flessibilità è evidenziata dalla sua capacità di soddisfare le esigenze [degli utenti] in continua evoluzione con il trascorrere del tempo. Si presupponga, ad esempio, che sia necessario archiviare dei nuovi files di grandi dimensioni e che un LV dedicato al file server non abbia spazio sufficiente per contenerli. Dato che non è stato utilizzato tutto lo spazio disponibile in `vg_critical`, sarà ancora possibile espandere `lv_files`. Il comando `lvresize` consente di estendere il volume logico, mentre il comando `resize2fs` consente di adattare il filesystem di conseguenza:

```

# df -h /srv/files/
Filesystem                      Size  Used   Avail   Use%  Mounted on
/dev/mapper/vg_critical-lv_files 5.00G  4.6G  146M   97%   /srv/files
# lvdisplay -C vg_critical/lv_files
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync
-> Convert
lv_files vg_critical -wi-ao-- 5.00g
# vgdiskdisplay -C vg_critical
      VG      #PV #LV #SN Attr   VSize VFree
      vg_critical 2    2    0 wz--n- 8.09g 2.09g
# lvresize -L 7G vg_critical/lv_files
Size of logical volume vg_critical/lv_files changed from 5.00 GiB (1280 extents)
to
-> 7.00GiB(1792extents).
Logical volume lv_files successfully resized
# lvdisplay -C vg_critical/lv_files
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync
-> Convert
lv_files vg_critical -wi-ao-- 7.00g
# resize2fs /dev/vg_critical/lv_files
resize2fs 1.42.12 (29-Aug-2014)

```

```

Filesystem at /dev/vg_critical/lv_files is mounted on /srv/files; on-line
resizing -> required
old_desc_blocks = 1, new_desc_blocks = 1
The filesystem on /dev/vg_critical/lv_files is now 1835008 (4k) blocks long.
# df -h /srv/files/
Filesystem Size Used Avail Use% Mounted on
/dev/mapper/vg_critical-lv_files 6.9G 4.6G 2.1G 70% /srv/files

```

ATTENZIONE

Come
ridimensionare i
filesystems

Non tutti i filesystems possono essere ridimensionati mentre sono attivi, pertanto dovete prima effettuare l'unmounting del file system, poi il ridimensionamento del filesystem e successivamente il mounting. È chiaro che per ridurre lo spazio destinato ad un LV, occorre prima ridurre il filesystem; diversamente se l'obiettivo è espandere il filesystem il ridimensionamento dovrà svolgersi invertendo le precedenti fasi ossia prima dovete estendere il volume logico e poi potrete espandere il filesystem. Quanto appena espresso è piuttosto ovvio: la dimensione del filesystem non potrà mai superare la dimensione del dispositivo a blocchi su cui si trova (sia che si tratti di una partizione fisica o di un volume logico). I filesystems ext3, ext4 e xfs possono essere estesi mentre sono ancora in funzione e senza unmounting; ma i summenzionati filesystems non possono essere ridotti senza unmounting. Invece il filesystem ReiserFS consente il ridimensionamento in espansione ed in riduzione senza unmounting. Il filesystem ext2, ormai datato, necessita dell'unmounting prima del ridimensionamento.

In teoria potreste procedere allo stesso modo allo scopo di estendere il volume che ospita il database, ma in questo caso il limite di capacità del VG è stato raggiunto:

```

# df -h /srv/base/
Filesystem          Size  Used  Avail Use% Mounted on
/dev/mapper/vg_critical-lv_base 1008M 854M 104M  90% /srv/base
# vgdisplay -C vg_critical
VG      #PV #LV #SN Attr   VSize VFree
vg_critical     2    2    0 wz--n- 8.09g 92.00m

```

Tutto ciò di fatto non è rilevante in quanto il sistema LVM consente anche di aggiungere volumi fisici a gruppi di volumi esistenti. Ad esempio, si prenda in considerazione la partizione sdb1 ovvero una partizione che non fa parte del sistema LVM e che contiene solo archivi che potrete trasferire volendo in lv_backups. Potrete di conseguenza destinarla ad altro, integrarla nel gruppo di volumi e così facendo recuperare lo spazio disponibile. Tutto ciò può essere messo in atto attraverso il comando vgextend. Ovviamente dovete preparare prima la partizione come volume fisico. Una volta che il VG è stato esteso, potrete utilizzare i comandi illustrati precedentemente per espandere prima il volume logico e poi il filesystem.

```

# pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created
# vgextend vg_critical /dev/sdb1
Volume group "vg_critical" successfully extended
# vgdisplay -C vg_critical
VG #PV #LV #SN Attr   VSize VFree
vg_critical     3    2    0 wz--n- 9.09g 1.09g
# [...]
[...]

```

```
# df -h /srv/base/
Filesystem           Size  Used Avail Use% Mounted on
/dev/mapper/vg_critical-lv_base 2.0G  854M  1.1G  45%  /srv/base
```

**ANDANDO
OLTRE**
**LVM - modalità
d'uso avanzate**

La tecnologia LVM soddisfa anche modalità d'uso più avanzate e potrete specificare diversi dettagli manualmente. Un amministratore sarà quindi in grado di effettuare il tweaking della capacità dei blocchi che compongono i volumi logici ed i volumi fisici, nonché il *physical layout* [trad. lett. la loro disposizione fisica]. Potrà anche trasferire i blocchi tra i PV, per esempio per il *fine-tuning* delle prestazioni oppure per ragioni pratiche come ad esempio per svuotare un PV per poter rimuovere il corrispondente disco fisico dal gruppo di volumi (eventualmente per assegnarlo ad un altro VG o per rimuoverlo definitivamente da un LVM). Le man pages che descrivono i comandi sono generalmente abbastanza chiare e dettagliate. Un buon “entry point” è la manual page lvm(8).

12.1.3. RAID o LVM?

I vantaggi delle tecnologie RAID e LVM sono innegabili a meno che non si abbia a che fare con un semplice computer desktop con installato un singolo hard disk le cui esigenze di modalità d'uso sono pressoché immutabili nel corso del tempo.

Tuttavia, le tecnologie RAID e LVM sono destinate a scopi diversi, ognuna con il rispettivo obiettivo ed è legittimo valutare quale dei due sistemi scegliere di adottare. La risposta dipenderà dai bisogni certi e previsti. In alcuni casi d'uso piuttosto modesti la questione non si pone. Ad esempio se desiderate proteggere i dati da guasti hardware, potrete optare per l'installazione di un RAID per poter configurare la ridondanza di un array di dischi, dato che la tecnologia LVM non è destinata a tale esigenza. Se, diversamente, desiderate uno storage scheme [trad. lett. un metodo di archiviazione] flessibile in cui i volumi sono indipendenti dal phisical layout dei dischi potrete optare per il sistema LVM dato che il RAID non è una scelta idonea.

NOTA
**Se per voi le
prestazioni
contano...**

Se per voi le prestazioni I/O (input/output) sono essenziali, soprattutto in termini di tempi di accesso, dovete considerare e confrontare i sistemi LVM e RAID con tutte le variabili d'uso nonché valutare l'effetto della vostra scelta. Solo in pochi casi d'uso le differenze di prestazioni saranno percettibili in quanto nella maggior parte dei casi saranno insignificanti. Potrete aver un maggior guadagno in termini di prestazioni attraverso l'uso di non-rotating storage media [in pratica e trad. non lett. i mezzi di trasmissione ed archiviazione di informazione che, a differenza degli hdd, CDs, ecc, non necessitano di una loro “rotazione” per la lettura dei dati] (come ad esempio i solid-state drives o SSDs); il loro costo in termini di megabyte è superiore rispetto quello dei dischi rigidi standard e di conseguenza la loro capacità è generalmente inferiore, ma le loro prestazioni sono eccellenti grazie al random access o direct access [Accesso casuale o diretto, è l'abilità di poter accedere entro gli stessi tempi di accesso a qualsiasi elemento arbitrario di una sequenza dati a prescindere dalla sua posizione in sequenza. Diversamente l'accesso sequenziale o sequential access non consente l'accesso a qualsiasi elemento arbitrario di una sequenza dati entro gli stessi tempi di accesso in quanto i tempi di accesso in sé dipendono dalla sua posizione.] Se la vostra modalità d'uso prevede diverse operazioni input/output in scattering (trad. lett. diffuse o disperse) su tutto il filesystem, ad esempio per databases in cui vengono eseguite frequenti complex queries gli SSDs vi offriranno dei benefici maggiori rispetto alla mera scelta fra LVM e RAID. Di conseguenza la vostra scelta non dovrebbe basarsi solo sulla pure speed (trad. lett. velocità effettiva), perché l'esigenza di prestazioni può essere facilmente soddisfatta dal semplice impiego di SSDs. [complex queries trad. lett. queries complesse, genericamente sono delle queries che non si basano solo su una relazione booleana fra oggetti bensì su altre particolarità come ad esempio la somiglianza qualitativa e quindi non solo sulla somiglianza quantitativa. La somiglianza qualitativa tra A e B viene descritta attraverso un isomorfismo, una funzione che associa tutti gli elementi di A con B, corrispondenti ad una data proprietà oppure ad una relazione o predicato]

Un altro caso d'uso, degno di nota, può essere rappresentato dall'esigenza di aggregare due dischi in un unico volume, dovuta a ragioni prestazionali o al bisogno di un singolo filesystem più vasto dei dischi disponibili. Il suddetto caso può essere soddisfatto configurando un array RAID-0 (o linear-RAID) oppure un volume LVM. A meno che non ci siano vincoli specifici (ad esempio per omogeneità in quanto gli altri computers utilizzano solo la tecnologia RAID), solitamente la scelta ricade sul sistema LVM.

La prima configurazione è leggermente più complessa, ma i costi iniziali sono tollerabili proprio per la flessibilità offerta successivamente, sia nel caso in cui dovessero cambiare i bisogni, sia nel caso in cui desideriate aggiungere dei dischi.

Infine potrebbe verificarsi un ulteriore caso d'uso, per certi versi più interessante, dove occorre conciliare resilienza ai guasti hardware e flessibilità nell'allocazione (ripartizione) dei volumi. I summenzionati sistemi sono insufficienti a soddisfare tali esigenze singolarmente; per ovviare si dovrà ricorrere ad entrambe le tecnologie contemporaneamente - o meglio, configurandole "una sull'altra". Il modello ormai di riferimento, dato che i sistemi RAID e LVM hanno ormai raggiunto una certa maturità, è mirato a garantire innanzitutto la ridondanza dei dati, aggregando i dischi in un numero ridotto di vasti RAID arrays per poi utilizzarli come volumi fisici LVM; le partizioni logiche per i filesystems saranno ricavate dai volumi logici (LVs). Il *selling point* di tale configurazione, qualora si guasti un disco, è rappresentato dalla possibilità di poter gestire solo un esiguo numero di RAID arrays per la ricostruzione, accorciando pertanto i tempi del recovery, messo in atto dall'amministratore. [*selling point* è un'espressione tecnica tipica del marketing, intraducibile letteralmente. Rappresenta l'unicità di un prodotto da reclamizzare in modo che gli acquirenti siano invogliati ad acquistarlo. Il suddetto termine tecnico non va confuso con l'espressione *bisogno indotto*, che diversamente rappresenta un bisogno non di prima necessità, ma "indotto" dalla pubblicità.]

Giusto per fare un esempio concreto si presupponga che: l'ufficio pubbliche relazioni alla Falcot Corp necessiti di una workstation per il video editing, ma che i vincoli di budget del suddetto ufficio non consentano di investire in hardware high-end e di fare riferimento ad una pianificazione bottom-up [Nella pianificazione bottom-up (dal basso verso l'alto) dettagliatamente si considerano gli elementi essenziali-funzionali sino a giungere in modo sequenziale all'obiettivo finale. Diversamente nella pianificazione top-down (dall'alto verso il basso) si considera l'obiettivo finale in correlazione agli indicatori economici, nazionali ed internazionali, suddividendo la pianificazione in sé in parti sempre più piccole sino a giungere agli elementi essenziali-funzionali. Tali termini vengono usati anche in informatica per lo sviluppo software e la programmazione, in economia, in filosofia, ecc.]. Si è scelto pertanto di privilegiare nella pianificazione l'hardware di natura specificatamente grafica (schermo e scheda video) e di utilizzare hardware generico per l'archiviazione. Tuttavia è alquanto risaputo che il digital video richieda dei requisiti di archiviazione ad hoc: la mole di dati da archiviare è piuttosto notevole e il throughput rate (trad. non lett. la velocità effettiva) di lettura e di scrittura dei dati si riflette sulle prestazioni dell'intero sistema (per fare un paragone conta più del tipico time access - tempo di accesso). [Per time access si intende genericamente l'arco di tempo che intercorre fra una richiesta inviata ad un dispositivo elettronico e la risposta da parte di quest'ultimo] Questi vincoli devono essere soddisfatti persino con hardware generico, ossia in questo caso da due hard disks SATA da 300 GB; inoltre si dovrà garantire la resilienza sia dei dati del sistema, sia degli utenti. Occorre garantire che il salvataggio dei videoclips esportati dopo l'editing sia protetto, mentre i video rushes (trad. non lett. flussi video), ancora in attesa dell'editing, sono meno critici, dato che vengono esportati direttamente dalle cassette video [o da nastri o da altri tipi di supporti per la memorizzazione di dati audio-video].

Per soddisfare i summenzionati requisiti come soluzione è stata scelta la combinazione delle tecnologie RAID-1 e LVM. I due dischi, denominati rispettivamente sda e sdc, sono stati collegati a due differenti controllers SATA al fine di ottimizzare l'accesso parallelo ai dispositivi e limitare il rischio di loro guasti simultanei. I suddetti dischi sono stati suddivisi in partizioni in base al seguente schema:

```
# fdisk -l /dev/sda
Disk /dev/sda: 300 GB, 300090728448 bytes, 586114704 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x00039a9f

Device      Boot   Start     End   Sectors   Size    Id     Type
/dev/sda1    *      2048  1992060  1990012   1.0G   fd    Linux raid autodetect
/dev/sda2        1992061 3984120  1992059   1.0G   82    Linux swap / Solaris
/dev/sda3        4000185 586099395 582099210 298G    5    Extended
/dev/sda5        4000185 203977305 199977120 102G   fd    Linux raid autodetect
/dev/sda6        203977306 403970490 199993184 102G   fd    Linux raid autodetect
/dev/sda7        403970491 586099395 182128904  93G   8e    Linux LVM
```

- Le prime partizioni di entrambi i dischi (circa 1 GB) sono state assemblate in un volume RAID-1, denominato md0. Questo mirror viene utilizzato direttamente per memorizzare il root filesystem.
- Le partizioni sda2 e sdc2 sono state impiegate come partizioni di swap, in modo da raggiungere 2 GB di spazio totale destinato allo swap. La workstation monta 1 GB di RAM, una quantità di memoria più che sufficiente.
- Sia le partizioni sda5 e sdc5, sia sda6 e sdc6 sono state assemblate in due nuovi volumi RAID-1 di circa 100 GB ciascuno, denominati rispettivamente md1 e md2. I suddetti mirrors sono stati inizializzati come volumi fisici LVM e assegnati al gruppo di volumi vg_raid. Così facendo tale gruppo di volumi può disporre di circa 200 GB di spazio protetto.
- Le restanti partizioni, sda7 e sdc7, sono state usate direttamente come volumi fisici ed assegnate ad un altro VG, denominato vg_bulk, che dispone di uno spazio di quasi 200 GB.

Una volta creati i VGs, potrete suddividerli in partizioni in modo flessibile. Dovrete semplicemente tenere a mente che i LVs realizzati in vg_raid verranno protetti da potenziali guasti dei dischi, diversamente dai LVs realizzati in vg_bulk; tuttavia, l'ultimo gruppo di volumi è stato distribuito in parallelo su ambedue i dischi, di conseguenza la velocità di lettura o di scrittura dei files di grandi dimensioni sarà elevata.

Creerete quindi i volumi logici lv_usr, lv_var, lv_home in vg_raid, per ospitare i corrispondenti filesystems; realizzerete anche il volume logico lv_movies per ospitare i film di cui è stato già stato effettuato l'editing. Nell'altro VG realizzerete un vasto volume logico, denominato lv_rushes, che ospiterà il flusso dati proveniente direttamente dalle videocamere digitali ed un volume logico denominato lv_tmp per i files temporanei. La posizione per la work area [trad. lett. area di lavoro, ossia lo spazio in cui gli utenti mettono in atto le loro attività ed in sostanza le ultimano] potrebbe far sorgere qualche perplessità: difatti le prestazioni hanno una loro importanza, ma il rischio di perdere quanto svolto per il guasto di un disco durante una sessione editing riesce ad essere giustificato dai benefici delle prestazioni? In base alla risposta a tale domanda, dovrete decidere dove creare un dato LV, se nel primo VG o nel secondo.

Il sopra descritto modello offre sia la ridondanza necessaria per i dati rilevanti, sia la flessibilità richiesta per la distribuzione dello spazio disponibile alle diverse applicazioni. Qualora abbiate bisogno di installare un nuovo software in un secondo momento (per l'editing di clips audio, ad esempio) potrete espandere senza conseguenze il volume logico che ospita /usr/.

NOTA
Perché tre
volumi RAID-1?

Nell'esempio soprastante in alternativa si poteva configurare un singolo volume RAID-1, da utilizzare come volume fisico per vg_raid. Perché allora sono stati creati tre volumi?

La prima suddivisione (md0 vs le altre) è stata ponderata per garantire la sicurezza dei dati: i dati scritti su entrambi gli elementi di un RAID-1 mirror sono identici, pertanto qualora fosse necessario sarà possibile montare uno dei dischi senza che il RAID layer ne risenta. Ad esempio nel caso in cui si verificasse un bug del kernel o i metadati LVM fossero corrotti, si potrà comunque avviare un sistema minimale per tentare di recuperare i dati critici tra cui il layout dei dischi nei volumi RAID e LVM; potrete in questo modo ricostruire i metadati ed accedere nuovamente ai files, così che il sistema possa essere ripristinato ad un suo nominal state (stato teorico).

La seconda suddivisione (md1 vs md2) è più soggettiva e dipende dalla consapevolezza che potrebbero sorgere degli eventi accidentali. Infatti non sempre le esigenze di storage sono chiare quando si assembla una workstation; inoltre le esigenze di storage potrebbero variare nel corso del tempo. In questo caso d'uso per esempio, è difficile conoscere in anticipo gli effettivi bisogni di archiviazione per i video rushes ed i video clip completi. Poniamo il caso che un dato video clip richieda una quantità molto elevata di video rushes e che il gruppo di volumi dedicato alla ridondanza dei dati sia occupato per meno del 50%, di conseguenza potrete considerare di recuperarne lo spazio non utilizzato.

Quindi potrete rimuovere uno dei volumi fisici, come md2, da vg_raid e riassegnarlo direttamente a vg_bulk (a condizione che l'operazione duri poco e che allo stesso tempo possiate continuare a svolgere le vostre mansioni con un conseguente calo di prestazioni), oppure annullare la configurazione RAID-1 su md2 ed integrare le componenti sda6 e sdc6 nel VG bulk (gruppo di volumi che viene espanso da 100 GB a 200 GB di spazio); poi potrete espandere il volume logico lv_rushes secondo le vostre necessità.

12.2 Virtualizzazione

La virtualizzazione è una delle più importanti invenzioni dell'informatica degli ultimi anni. La sua denominazione deriva dalle diverse astrazioni e tecniche che consentono di simulare macchine virtuali con un margine variabile di indipendenza dall'hardware effettivo. Grazie a tale tecnologia potrete ospitare su un unico server fisico diversi sistemi in funzione contemporaneamente ed isolatati fra loro. Potrete impiegare la suddetta tecnologia per diverse attività, che spesso necessitano del dovuto isolamento: test environments [server virtuali] per verificare diverse configurazioni, oppure per gestire in sicurezza i diversi hosted services [i servizi implementati da macchine diverse rispetto alle macchine "fisicamente" destinate a tali servizi, generalmente tramite internet] separandoli su macchine virtuali differenti.

Esistono molteplici soluzioni per la virtualizzazione, ciascuna con vantaggi e svantaggi. Questo manuale si concentrerà solo su Xen, LXC e KVM, ma occorre citare altre notevoli implementazioni:

- QEMU è un software emulator [un emulatore hardware per software] in grado di virtualizzare le specifiche complete di un computer; sebbene le prestazioni siano inferiori di velocità, rispetto alle prestazioni dell'hardware reale, questo emulatore hardware consente di testare sistemi operativi [non nativi] senza necessità di modificarli e persino sistemi operativi sperimentali. Potrete pertanto emulare un'un'architettura hardware diversa: ad esempio, potrete emulare con un computer amd64 l'architettura arm. QEMU è un free software.

♦ <http://www.qemu.org/>

- Bochs è un'altra macchina virtuale gratuita ma emula solo architetture x86 (i386 e amd64).
- VMWare è una macchina virtuale con diritti di proprietà intellettuale. Si tratta del primo fra gli emulatori pubblicati e di conseguenza è anche uno dei più conosciuti. Si basa su principi simili a QEMU. Dispone inoltre di funzionalità avanzate come ad esempio lo snapshotting [trad. lett. "acquisizione di un'istantanea", in analogia all'istantanea fotografica] dello stato di una macchina virtuale in esecuzione [molto genericamente serve per il ripristino di un dato stato del sistema emulato].

♦ <http://www.vmware.com>

- VirtualBox è una macchina virtuale perlopiù gratuita (sebbene alcuni componenti aggiuntivi siano disponibili con una licenza con diritti di proprietà intellettuale). Sfortunatamente VirtualBox si trova nella sezione "contrib", perché include alcuni files precompilati che non possono essere ricostruiti (rebuild) senza un compilatore "proprietario". Questo emulatore è più recente rispetto VMWare e si limita alla virtualizzazione delle architetture i386 e amd64, ma include alcune funzionalità pregevoli come lo snapshotting.

♦ <http://www.virtualbox.org/>

12.2.1. Xen

Xen è una soluzione di "paravirtualizzazione" [da non confondersi con la virtualizzazione]. In pratica tale tecnologia prevede la frapposizione di un sottile abstraction layer [trad. lett. layer (livello) di astrazione - vedi modello ISO/OSI] tra l'hardware e gli upper system [i sistemi operativi gerarchicamente soprastanti nella scala del modello ISO/OSI], denominato "hypervisor" (trad. lett. ipervisore), il cui ruolo è simile agli arbiters [i dispositivi elettronici che si occupano della distribuzione delle risorse condivise] ossia gestisce l'accesso all'hardware dalle macchine virtuali.

Tuttavia, l'hypervisor di fatto riceve solo poche istruzioni, in quanto l'hardware risponde direttamente alle richieste dei sistemi. Il principale pregio di Xen è che le prestazioni non vengono degradate ed i sistemi processano con prestazioni in termini di velocità molto simili alle prestazioni delle specifiche hardware; lo svantaggio di Xen è che dovete modificare i kernels dei sistemi operativi per renderli compatibili all'hypervisor di Xen e poterli infine eseguire con Xen.

In termini tecnici. L'hypervisor è il layer più basso in esecuzione direttamente sull'hardware e sottostante il layer del kernel. L'hypervisor in sostanza divide il resto del software fra i diversi domains (domini), corrispondenti ad altrettante macchine virtuali. Uno di questi domini (il primo ad essere avviato) denominato dom0, riveste un compito particolare, in quanto solo da questo dominio è possibile gestire l'hypervisor e la messa in esecuzione degli altri domini. Gli altri domini sono denominati domU. In pratica e dal punto di vista di qualsiasi utente il dominio "dom0" corrisponde all'"host" su cui vengono virtualizzati gli altri sistemi "domU" corrispondenti ai "guests".

CULTURA Xen e le diverse versioni di Linux

In origine Xen è stato sviluppato come un insieme di patches non canoniche e non integrate nel kernel ufficiale. Nel frattempo sono stati distribuiti diversi sistemi di virtualizzazione emergenti (in particolare KVM) che necessitavano di diverse funzionalità generiche di virtualizzazione e di conseguenza, per soddisfare tali esigenze, nel kernel Linux sono state introdotte un set di funzionalità (l'interfaccia nota come `paravirt_ops` o `pv_ops`). Ma le patches di Xen dell'epoca non potevano essere accettate ufficialmente dato che erano sostanzialmente il doppione di alcune delle funzionalità della summenzionata interfaccia.

La Xensource , la società dietro Xen, ha deciso quindi di trasferire il suo prodotto nel suddetto framework, in modo che potesse essere integrato ufficialmente nel kernel Linux. Pertanto tale società ha riscritto gran parte del codice e per quanto abbia realizzato una versione di Xen funzionante e basata sull'interfaccia `paravirt_ops`, la conseguente integrazione nel kernel ufficiale di Linux è stata molto graduale. Difatti l'integrazione è stata completata con Linux 3.0.

- ♦ <http://wiki.xenproject.org/wiki/XenParavirtOps>

Dato che Jessie si basa sulla versione 3.16 del kernel Linux, le immagini standard `linux-image-686-pae` e `linux-image-amd64` includono già il codice necessario diversamente dalle versioni precedenti di Debian, da Squeeze in poi, che richiedono l'integrazione del patching specifico in base alla distribuzione.

- ♦ http://wiki.xenproject.org/wiki/Xen_Kernel_Feature_Matrix

CULTURA Xen ed i kernels non di Linux

Xen richiede che i sistemi operativi da eseguire su tramite debbano essere modificati e personalizzati ad hoc; purtroppo non tutti i kernels hanno raggiunto lo stesso livello di maturità per tale esigenza. Alcuni kernels sono pienamente funzionanti, sia come dom0, sia come domU: Linux 3.0 e le versioni successive; NetBSD 4.0 e le versioni successive; OpenSolaris. Altri kernels attualmente funzionano solo come domU. Potrete verificare la compatibilità di ciascun sistema operativo in [wiki Xen](#):

- ♦ http://wiki.xenproject.org/wiki/Dom0_Kernels_for_Xen
- ♦ http://wiki.xenproject.org/wiki/DomU_Support_for_Xen

In conclusione Xen può contare sulle funzionalità hardware specificamente dedicate alla virtualizzazione (presenti solo nei processori più recenti), ma può eseguire i sistemi operativi non personalizzati ad hoc solo come domU (incluso Windows).

NOTA Architetture compatibili con Xen

Xen è attualmente disponibile solo per architetture i386, amd64, arm64 e armhf

Per usare Xen sotto Debian, sono necessari tre componenti:

- L'hypervisor, il cui correlato pacchetto, dovrà essere scelto in base all'hardware: `xen-hypervisor-4.4-amd64`, `xen-hypervisor-4.4-armhf` o `xen-hypervisor-4.4-arm64`.
- Un kernel compatibile con l'hypervisor. Ossia qualsiasi kernel a partire dalla versione 3.0 e di conseguenza inclusa la versione 3.16 presente in Jessie.
- L'architettura i386 necessita di una libreria standard ad hoc resa disponibile attraverso patches adeguate per poter utilizzare Xen; tali patches sono distribuite attraverso il pacchetto `libc6-xen`.

Per evitare di dover selezionare i suddetti componenti manualmente potrete utilizzare uno dei metapacchetti di supporto disponibili (come ad esempio `xen-linux-system-amd64`); questi metapacchetti fanno riferimento alle combinazioni di hypervisor e versioni del kernel più conosciute. L'hypervisor vi consiglierà anche il pacchetto `xen-utils-4.4`, che include a sua volta gli strumenti per gestire l'hypervisor da dom0. `xen-utils-4.4` vi consiglierà le librerie standard appropriate. L'installazione in sé, attraverso i configuration scripts [trad. lett. "scripts di configurazione"], determina la creazione di una nuova voce nel menu del bootloader Grub, consentendovi successivamente di avviare il kernel scelto per uno Xen dom0. Si precisa ovviamente che questa nuova voce non sarà la prima nell'elenco delle voci del bootloader e di conseguenza il correlato processo non sarà avviato automaticamente di default. Qualora desideriate diversamente potrete eseguire i sottostanti comandi:

```
# mv /etc/grub.d/20_linux_xen /etc/grub.d/09_linux_xen  
# update-grub
```

Una volta che avrete installato i prerequisiti, dovrete testare il corretto funzionamento di dom0, riavviando l'hypervisor ed il kernel Xen. Il sistema dovrebbe avviarsi senza problemi ed appariranno nella console alcuni avvisi aggiuntivi durante le prime fasi di inizializzazione.

Giunti a questo passo potrete installare i sistemi operativi sui sistemi domU, utilizzando gli strumenti di `xen-tools`. Questo pacchetto include il comando `xen-create-image`, che automatizza le tasks. Il suo unico parametro obbligatorio è `--hostname` e conferisce un nome a domU; sono importanti anche altre opzioni, ma non dovrete obbligatoriamente impostarle da riga di comando perché potrete farlo senza causare errori direttamente attraverso il file di configurazione `/etc/xen-tools/xen-tools.conf`. Avrete quindi due opportunità: verificare il contenuto del sopracitato file prima di creare immagini; oppure potrete trasmettere i parametri aggiuntivi attraverso il comando `xen-create-image`. I parametri in particolare degni di menzione sono i seguenti:

- `--memory`, specifica la quantità di RAM da dedicare al sistema creato;
- `--size` e `--swap`, definiscono la capacità dei "dischi virtuali" per domU;
- `--debootstrap`, specifica che il sistema deve essere installato con `debootstrap`; con questa opzione dovrete usare anche `--dist` (per indicare il nome della distribuzione, come ad esempio `jessie`).
- `--dhcp` impone che la configurazione della rete domU sia ottenuta tramite DHCP; invece, `--ip` consente di definire un indirizzo IP statico.
- Infine, dovrete scegliere un metodo di archiviazione per le immagini che dovete creare (verranno considerate come dischi rigidi in domU). Il metodo più semplice corrisponde all'opzione `--dir` e consiste nel creare un file su dom0 per ciascun dispositivo che si desidera concedere a domU.

Invece sui sistemi che utilizzano LVM potrete utilizzare l'opzione `--lvm` seguita dal nome di un gruppo di volumi; dopodiché `xen-create-image` creerà un nuovo volume logico all'interno del gruppo di volumi precedentemente indicato e questo stesso volume logico sarà messo a disposizione di domU come hard disk.

NOTA

Come archiviare in domU

Potrete esportare su domU interi hard disks nonché partizioni, RAID arrays o volumi logici LVM preesistenti. Tali operazioni purtroppo non vengono automatizzate da `xen-create-image` e quindi dovete modificare manualmente il file di configurazione dell'immagine Xen, dopo la creazione dell'immagine in sé attraverso `xen-create-image`.

ANDANDO OLTRE

Come installare i sistemi diversi da Debian in un domU

Qualora abbiate intenzione di installare un sistema non Linux, ricordatevi di specificare il kernel che deve essere utilizzato da domU tramite l'opzione `--kernel`.

Al termine della suddetta configurazione, potrete creare l'immagine del vostro futuro Xen domU:

```
# xen-create-image --hostname testxen --dhcp --dir /srv/testxen --size=2G --dist=> jessie --role=udev
```

```
[...]
General Information
-----
Hostname      : testxen
Distribution   : jessie
Mirror: http://ftp.debian.org/debian/
Partitions: swap           128Mb (swap)
             /              2G     (ext3)

Image type : sparse
Memory size: 128Mb
Kernel path: /boot/vmlinuz-3.16.0-4-amd64
Initrd path: /boot/initrd.img-3.16.0-4-amd64
[...]
LogFile produced at:
    /var/log/xen-tools/testxen.log
```

```
Installation Summary
-----
Hostname : testxen
Distribution : jessie
MAC Address : 00:16:3E:8E:67:5C
IP-Address(es) : dynamic
RSA Fingerprint : 0a:6e:71:98:95:46:64:ec:80:37:63:18:73:04:dd:2b
Root Password : Root Password:adaX2jyRHnUWm8BDJS7PcEJ
```

La macchina virtuale è stata creata, ma di fatto è spenta e lo spazio dell'hard disk concretamente in uso è quello riservato a dom0. Ovviamente potrete creare diverse immagini, con parametri diversi in base alle vostre necessità.

Ma dovete stabilire anche la modalità d'accesso, prima di poter mettere in esecuzione le suddette macchine virtuali. Sarà necessario decidere se utilizzarle come macchine isolate, accessibili soltanto dalla loro system console [trad. lett. console di sistema ossia l'interfaccia testuale dopo aver effettuato l'accesso al sistema tramite il system logger], sebbene tale modello d'uso venga raramente adottato. Oppure dovete stabilire se utilizzare le domU alla stregua di server remoti accessibili attraverso la rete. Tuttavia non è possibile associare una scheda di rete per ciascuna domU; di conseguenza Xen permette quindi di creare delle interfacce virtuali, in modo che ciascun

dominio possa essere rilevato attraverso la modalità standard. In ogni caso per poter impiegare le schede di rete virtuali dovete connetterle ad una rete, anche se virtuale. Xen supporta diversi modelli (o modalità) di rete:

- il modello **bridge**, è il modello di rete più semplice fra tutti; in pratica le schede di rete eth0 funzionano (sia con dom0, sia con domU) come se fossero collegate direttamente ad uno switch di rete Ethernet.
- il modello **routing**, in cui dom0 funge da router frapposto tra i sistemi domU e la rete esterna (fisica);
- il modello **NAT**, acronimo di **network address translation** in inglese [traduzione degli indirizzi di rete (NAT) in italiano], in cui dom0 è frapposto nuovamente tra i sistemi domU ed il resto della rete, tuttavia i sistemi domU non sono accessibili direttamente dall'esterno, dato che il traffico deve prima superare il NAT su dom0.

Queste tre networking nodes [ovvero in questo caso punti di distribuzione] mettono a disposizione diverse interfacce con nomi insoliti, come `vif*`, `veth*`, `peth*` e `xenbr0`. Lo Xen hypervisor assegna tali interfacce in base al layout stabilito, sotto il controllo dei tools dello spazio utente. I modelli NAT e routing non verranno trattati in questa sede dato che vengono impiegati solo in casi particolari, ma verrà preso in esame soltanto il modello bridge.

La configurazione standard dei pacchetti di Xen non apporta alcuna modifica alla configurazione di rete del sistema. D'altra parte, il demone `xend` viene configurato per integrare le interfacce di rete virtuali in qualsiasi bridge di rete preesistente (viene utilizzato `xenbr0` qualora ci fossero diversi bridges). Dovrete pertanto configurare un bridge in `/etc/network/interfaces` sostituendo la voce `eth0` (dopo aver installato il pacchetto `bridge-utils`, raccomandato a sua volta, proprio per tale fine, dal pacchetto `xen-utils-4.4`):

```
auto xenbr0
iface xenbr0 inet dhcp
    bridge_ports eth0
    bridge_maxwait 0
```

Dopo il riavvio, per verificare se il bridge è stato creato automaticamente, potrete avviare domU utilizzando gli strumenti di controllo di Xen, in particolare il comando `xl`. Quest'ultimo comando permette di eseguire varie operazioni sui domini, tra cui la loro visualizzazione in formato elenco, il loro avvio/spegnimento.

```
# xl list
Name                           ID  Mem  VCPUs  State   Time(s)
Domain-0                        0   463      1  r-----  9.8
# xl create /etc/xen/testxen.cfg
Parsing config from /etc/xen/testxen.cfg
# xl list
Name                           ID  Mem  VCPUs  State   Time(s)
Domain-0                        0   366      1  r-----  11.4
testxen                         1   128      1  -b----  1.1
```

STRUMENTI TOOLS

Selezione dei
toolstacks per
gestire Xen VM

Fino alla versione di Debian 7, `xm` era lo strumento da riga di comando di riferimento per l'utilizzo e la gestione delle macchine virtuali Xen. È stato sostituito da `xl`, che è nella maggior parte dei casi compatibile. Ma questi non sono gli unici strumenti disponibili: come strumenti alternativi potrete usare `virsh` di `libvirt` e `xe` di XAPI, che a sua volta appartiene a XenServer (una versione commerciale di Xen).

[Il toolstack è una collezione di strumenti per lavorare su diverse componenti di un correlato sistema]

ATTENZIONE
Ad ogni
immagine potrà
essere assegnato
un solo domU!

Potrete avviare più sistemi domU in parallelo, ma ciascun sistema domU dovrà utilizzare un'immagine dedicata, in quanto in generale un sistema domU processa ritenendosi l'unico sistema installato sull'hardware (tranne la piccola porzione del kernel che interagisce con l'hypervisor); due sistemi domU che processano contemporaneamente non possono condividere lo stesso spazio di archiviazione. Tuttavia i sopraccitati sistemi domU, se non vengono impiegati per processare contemporaneamente, potranno usufruire della stessa partizione di swap o della stessa partizione dedicata all'hosting del filesystem /home.

Si precisa che la domU testxen occupa RAM reale, ovvero quella RAM che diversamente sarebbe stata destinata a dom0 (quindi non si tratta di RAM simulata). Pertanto se avete intenzione di costruire un server destinato ad ospitare le istanze di Xen dovete calcolarne la RAM appositamente.

E voilà, la macchina virtuale è in grado di avviarsi! Giunti a questo punto potrete accedervi in due modi. Il primo metodo, abbastanza comune, prevede la connessione "da remoto" attraverso la rete, come se si trattasse di una macchina reale, e di conseguenza richiederà la configurazione di un server DHCP oppure di un DNS. Il secondo metodo potrebbe rivelarsi utile qualora la configurazione della rete domU non andasse a buon fine, e prevede l'impiego della console hvc0 attraverso il comando `xl console`:

```
# xl console testxen
[...]
Debian GNU/Linux 8 testxen hvc0

testxen login:
```

Ciò vi consente di aprire una sessione, come se foste fisicamente di fronte alla tastiera della macchina virtuale. Per scollarvi dalla console dovete utilizzare la combinazione di tasti `Control +]`.

SUGGERIMENTO
Come ottenere la
console in modo
diretto

Se desiderate avviare un sistema domU ed accedere alla sua console in modo diretto, dovete trasmetterne lo switch -c al comando `xl create`. L'avvio di un sistema domU attraverso il summenzionato switch comporterà la visualizzazione di tutti i messaggi durante gli avvii del sistema.

STRUMENTI TOOLS
OpenXenManager

OpenXenManager (incluso nel pacchetto `openxenmanager`) è un'interfaccia grafica che consente la gestione remota dei domini Xen grazie all'ausilio dell'API Xen. Ossia la gestione da remoto dei domini Xen. Supporta la maggior parte delle funzionalità del comando `xl`.

Quando attiverete il domU questi potrà essere utilizzato come qualsiasi altro server (dopo tutto si tratta di un sistema GNU/Linux). Inoltre lo status di macchina virtuale conferisce al suddetto sistema alcune funzionalità aggiuntive. Potrete ad esempio metterlo in pausa temporaneamente e poi riattivarlo, attraverso i comandi `xl pause` e `xl unpause`. Occorre precisare che un sistema domU in pausa smette di consumare la potenza del processore, ma continua ad usufruire della sua memoria allocata. Di conseguenza le funzionalità di backup (`xl save`) e del correlato ripristino (`xl restore`) potrebbero essere delle alternative migliori: difatti il salvataggio-backup di un sistema domU libera le risorse utilizzate dallo stesso sistema domU, compresa la RAM. Il sistema domU quando viene ripristinato (o riattivato dopo una pausa) tiene in considerazione solo il tempo trascorso dalla sua inattività e nient'altro. Qualora venisse improvvisamente spento un sistema dom0 mentre è ancora in esecuzione un sistema domU, gli scripts del pacchetto automaticamente avvieranno la funzionalità di backup del sistema domU in questione, in modo da poterlo ripristinare al successivo avvio.

Naturalmente, quanto appena illustrato comporterà gli stessi inconvenienti riscontrabili ad esempio con un computer portatile in ibernazione; in particolare se la sospensione del sistema domU dura troppo tempo le connessioni di rete potrebbero "scadere". Occorre notare che Xen finora è perlopiù incompatibile con la gestione dell'alimentazione ACPI, in quanto quest'ultima non consente la sospensione software del sistema host (dom0).

**DOCUMENTAZIONE
opzioni del comando
xl**

La maggior parte dei sottocomandi `xl` esige uno o più argomenti, tra cui spesso persino il nome della domU interessata. I suddetti argomenti sono elencati nella man page `xl(1)`.

Potrete spegnere o riavviare un sistema domU dallo stesso sistema (eseguendo il comando `shutdown`) oppure dal sistema dom0 attraverso `xl shutdown` o `xl reboot`.

**ANDANDO OLTRE
Xen (uso avanzato)**

Xen comprende diverse funzionalità che non possono essere descritte in pochi paragrafi. Difatti si tratta di un sistema piuttosto dinamico che consente di gestire diversi parametri per un mero dominio (la quantità di memoria allocata, i dischi rigidi da rendere disponibili, il funzionamento del task scheduler, ecc.) addirittura mentre il summenzionato dominio è in esecuzione. Un domU può persino essere migrato tra servers, senza necessità di essere spento e senza alcun rischio di perdere le connessioni di rete! La miglior fonte di informazioni per tutti questi aspetti (avanzati) è la documentazione ufficiale di Xen.

♦<http://www.xen.org/support/documentation.html>

12.2.2. LXC

LXC non può essere definito strettamente parlando un sistema di virtualizzazione, nonostante suo tramite sia possibile costruire "macchine virtuali", dato che in realtà si tratta di un sistema per isolare fra loro gruppi di processi in esecuzione sullo stesso host. LXC si basa su una serie di recenti sviluppi nel kernel Linux, denominati collettivamente control groups, attraverso cui una collezione di processi chiamati "groups" possono prendere visione di determinati aspetti dell'intero sistema. Fra questi aspetti sono degni di nota i "process identifiers" [trad. lett. gli identificatori di processo], la configurazione di rete ed i mount points [trad. lett. punti di montaggio]. In questo modo un "gruppo di processi isolato" non avrà accesso ad altri processi nel sistema, ma esclusivamente ad uno specifico subset del file system predefinito. Inoltre un "gruppo di processi isolato" possiede un'interfaccia di rete ed una routing table [o tabella di routing o tabella di instradamento] dedicate, e può eventualmente essere configurato per rilevare soltanto un sottoinsieme delle periferiche disponibili nel sistema.

Le summenzionate funzionalità possono essere combinate per isolare un'intera famiglia di processi sin dal processo `init` e l'insieme dei processi risultanti è simile ad una macchina virtuale. Tale configurazione viene denominata ufficialmente "container" (dato che LXC è l'acronimo di Linux Containers), ma diversamente da una macchina virtuale Xen o KVM un container non include un secondo kernel; ossia un container impiega "quasi" lo stesso kernel del sistema host. Questo aspetto consente dei vantaggi ma introduce anche dei svantaggi: in particolare tra i vantaggi occorre citare sia l'assenza di overhead [attività accessorie necessarie per mettere in esecuzione una data attività] che migliora le prestazioni, sia il fatto che il kernel rileva, attraverso un quadro generale e non parziale, i processi del sistema, rendendo lo scheduling più efficiente rispetto allo scheduling di una serie di attività differenti attuato da due kernels distinti. Uno dei svantaggi principali è che non è possibile mettere in esecuzione un altro kernel nel container (ovvero né un sistema operativo completamente differente, né un'altra versione di Linux).

NOTA I limiti dell'isolamento di LXC	<p>Gli LXC containers non offrono lo stesso livello di isolamento raggiunto dai più gravosi emulatori o virtualizzatori. In particolare:</p> <ul style="list-style-type: none"> • dal momento che il kernel è condiviso tra il sistema host ed i containers, i processi legati ai containers possono comunque accedere ai messaggi del kernel e qualora tali messaggi venissero trasmessi dal contenitore ciò comporterebbe una grave perdita di informazioni. • sempre per la summenzionata causa, qualora uno dei contenitori fosse compromesso potrebbe essere sfruttata una vulnerabilità del kernel per infettare anche gli altri contenitori. • per quanto riguarda il file system il kernel verifica i permessi in base agli identificatori numerici degli utenti [UID] e dei gruppi [GID]; dovrete fare pertanto attenzione alle parti del filesystem con permessi di scrittura condivise fra i containers, visto che gli identificatori si basano su ogni singolo container e potrebbero individuare svariati utenti e gruppi.
---	--

Dato che in sostanza si tratta di un isolamento e non di una virtualizzazione piena, la configurazione dei contenitori LXC è un po' più complessa rispetto alla semplice messa in esecuzione di debian-installer su una macchina virtuale. A seguire verranno descritti i prerequisiti e poi come impostare una configurazione di rete; così facendo sarete effettivamente in grado di creare il sistema che verrà eseguito dal container.

12.2.2.1 Le fasi preliminari

Dovrete installare il pacchetto lxc che include gli strumenti necessari per eseguire LXC. LXC necessita anche del sistema di configurazione dei control groups, che di fatto è un virtual filesystem montato in /sys/fs/cgroup. Dato che Debian 8 è passato a systemd, che a sua volta si basa sui control groups, tutto ciò viene implementato automaticamente all'avvio senza ulteriori configurazioni.

12.2.2.2 Configurazione di rete

L'obiettivo dell'installazione di LXC è la configurazione di macchine virtuali; potrete pertanto decidere di lasciare isolate dalla rete le suddette macchine e di comunicare con loro solo tramite il filesystem, ma sappiate che nella maggior parte dei casi d'uso è opportuno concedere ai containers un accesso minimo alla rete. Uno dei casi tipici prevede che ciascun contenitore abbia un'interfaccia di rete virtuale e che la connessione alla rete effettiva avvenga tramite un bridge. La suddetta interfaccia virtuale può essere collegata direttamente all'interfaccia fisica di rete della macchina host (per connettere il contenitore direttamente alla rete) oppure ad un'altra interfaccia virtuale individuata dall'host (per far filtrare o instradare il traffico dall'host). In entrambi i casi, sarà necessario installare il pacchetto bridge-utils.

Per fare tutto ciò dovete solamente modificare /etc/network/interfaces, sostituendo la configurazione dell'interfaccia fisica (per esempio eth0) con quella dell'interfaccia bridge (solitamente br0) ed aggiungendo il collegamento tra le due interfacce. Quindi poniamo ad esempio che il configuration file delle interfacce di rete inizialmente contiene le voci seguenti:

```
auto eth0
iface eth0 inet dhcp
```

Dovrete disattivarle e sostituirle con:

```
#auto eth0
#iface eth0 inet dhcp

auto br0
iface br0 inet dhcp
    bridge-ports eth0
```

L'efficacia della soprastante configurazione sarà simile a quella che si potrebbe ottenere dalla configurazione di una connessione diretta dei containers all'interfaccia di rete fisica dell'host. La configurazione "bridge" gestisce il transito dei frames Ethernet [ovvero le unità dati dei pacchetti di rete] fra tutte le interfacce collegate al bridge, pertanto sia l'interfaccia fisica `eth0`, sia le interfacce stabilite per i contenitori.

Se non è possibile utilizzare questa configurazione (ad esempio perché non è possibile assegnare un indirizzo IP pubblico ai containers), basterà creare un'interfaccia tap virtuale e connetterla al bridge [In generale le interfacce TUN e TAP sono dei dispositivi di rete virtuali che si occupano rispettivamente: l'interfaccia TUN del transito dei pacchetti IP (layer 3 del modello ISO/OSI); l'interfaccia TAP del transito dei frames Ethernet (layer 2 del modello ISO/OSI). Per quanto possano essere entrambe impiegate per il tunneling non possono essere utilizzate contemporaneamente in quanto operano su layer differenti] Come effetto di quanto sopra esposto la topologia di rete scaturente diventerà simile a quella di un host che possiede una seconda scheda di rete collegata ad uno switch separato in cui sono connessi i contenitori. L'host dunque deve fungere da gateway per i containers affinché possano comunicare con l'esterno. [La topologia di rete è un modello che si basa sulla teoria dei grafi e che rappresenta le connessioni logiche e fisiche dei nodi] Per ottenere di fatto la summenzionata configurazione occorrerà installare oltre a `bridge-utils` anche il pacchetto `vde2`; il file `/etc/network/interfaces` dovrà essere modificato di conseguenza:

```
# Interface eth0 is unchanged
auto eth0
iface eth0 inet dhcp

# Virtual interface
auto tap0
iface tap0 inet manual
    vde2-switch -t tap0

# Bridge for containers
auto br0
iface br0 inet static
    bridge-ports tap0
    address 10.0.0.1
    netmask 255.255.255.0
```

Potrete configurare la rete in modo statico nei contenitori oppure dinamico attraverso un DHCP server che processa nell'host. Però il DHCP server dovrà essere a sua volta configurato per rispondere alle queries sull'interfaccia `br0`.

12.2.3 Configurazione del sistema

Ora è arrivato il momento di configurare il filesystem che verrà utilizzato dal contenitore. Dal momento che questa "macchina virtuale" non verrà eseguita direttamente dall'hardware, sono necessari alcuni tweaks che sostanzialmente differenzieranno il suo filesystem rispetto ad uno tradizionale, soprattutto per quanto riguarda il kernel, le periferiche e le consoles. [Genericamente i tweaks sono un dosaggio raffinato (o fine-tuning) dei perfezionamenti di un sistema complesso come ad esempio un dispositivo elettronico o come in questo caso un filesystem] Fortunatamente, il pacchetto `lxc` contiene già gli scripts in grado di automatizzare ampiamente questa configurazione.

Ad esempio i comandi descritti a seguire (che necessitano dei pacchetti `debootstrap` e `rsync`) possono installare un Debian container:

```
root@mirwiz:~# lxc-create -n testlxc -t debian
debootstrap is /usr/sbin/debootstrap
Checking cache download in /var/cache/lxc/debian/rootfs-jessie-amd64 ...
Downloading debian minimal ...
I: Retrieving Release
I: Retrieving Release.gpg
[...]
Download complete.
Copying rootfs to /var/lib/lxc/testlxc/rootfs...
[...]
Root password is 'sSiKhMzI', please change !
root@mirwiz:~#
```

Se notate il filesystem viene inizialmente generato in `/var/cache/lxc` e poi trasferito nella directory di destinazione. Ciò consente di creare dei containers identici molto più velocemente, dato che sarà necessario effettuarne semplicemente la copia.

Si precisa che lo script di creazione del modello Debian permette l'immissione delle opzioni: `--arch` (in modo che si possa specificare l'architettura del sistema da installare) e `--release` (se si desidera una versione alternativa alla versione stabile corrente). Potrete anche configurare la variabile d'ambiente `MIRROR` per impostare un mirror Debian locale.

Il filesystem generato ex-novo contiene un sistema minimale di Debian e, per impostazione predefinita, il contenitore correlato non possiede un'interfaccia di rete (a parte un loopback). Visto che tale configurazione non corrisponde a quella auspicata, dovete modificare il file di configurazione del contenitore (`/var/lib/lxc/testlxc/config`) ed aggiungere le seguenti voci `lxc.network.*`:

```
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = br0
lxc.network.hwaddr = 4a:49:43:49:79:20
```

Queste righe indicano rispettivamente che: nel container verrà creata un'interfaccia virtuale; l'interfaccia virtuale verrà attivata automaticamente all'avvio dello stesso container; l'interfaccia virtuale si collegherà automaticamente al bridge `br0` dell'host; l'indirizzo MAC specifico dell'interfaccia virtuale. Dovrete disabilitare o rimuovere quest'ultima voce, se intendete utilizzare un indirizzo MAC casuale e generato random.

Infine potrete impostare come ultima voce l'hostname:

```
lxc.utsname = testlxc
```

12.2.2.4 Avvio del container

Ora che l'immagine della vostra macchina virtuale è pronta, potrete avviare il contenitore:

```

root@mirwiz:~# lxc-start --daemon --name=testlxc
root@mirwiz:~# lxc-console -n testlxc
Debian GNU/Linux 8 testlxc tty1

testlxc login: root
Password:
Linux testlxc 3.16.0-4-amd64 #1 SMP Debian 3.16.7-ckt11-1 (2015-05-24) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@testlxc:~# ps auxwf
USER     PID   %CPU   %MEM    VSZ   RSS   TTY      STAT START   TIME  COMMAND
root       1     0.0     0.2  28164  4431 ?        Ss  17:33  0:00  /sbin/init
root      20     0.0     0.1  32960  3160 ?        Ss  17:33  0:00  /lib/systemd/systemd-
journalald
root      82     0.0     0.3  55164  5456 ?        Ss  17:34  0:00  /usr/sbin/sshd -D
root      87     0.0     0.1  12656  1924 tty2      Ss+ 17:34  0:00  /sbin/agetty --noclear tty2
-> linux
root      88     0.0     0.1  12656  1764 tty3      Ss+ 17:34  0:00  /sbin/agetty --noclear tty3
root      89     0.0     0.1  12656  1908 tty4      Ss+ 17:34  0:00  /sbin/agetty --noclear tty4
-> linux
root      90     0.0     0.1  63300  2944 tty1      Ss  17:34  0:00  /bin/login --
root     117     0.0     0.2  21828  3668 tty1      S  17:35  0:00  \_ -bash
root     268     0.0     0.1  19088  2572 tty1      R+ 17:39  0:00  \_ ps auxfw
root     91     0.0     0.1  14228  2356 console  Ss+ 17:34  0:00  /sbin/agetty --noclear --
keep- --> baud console 115200 38400 9600 vt102
root     197     0.0     0.4  25384  7640 ?        Ss  17:38  0:00  dhclient -v -pf /run/
dhclient. ->eth0.pid -lf /var/lib/dhcp/dhclient.e
root     266     0.0     0.1  12656  1840 ?        Ss  17:39  0:00  /sbin/agetty --noclear
tty5
root     266     0.0     0.1  12656  1840 ?        Ss  17:39  0:00  /sbin/agetty --noclear
tty6
root@testlxc:~#

```

Giunti finalmente nel container, potrete accedere soltanto a: quei processi avviati dallo stesso container; al subset (dedicato al container) dell'intero filesystem (`/var/lib/lxc/testlxc/rootfs`). Potrete uscire dalla console attraverso la combinazione dei tasti `Control + a q`.

Potrete notare che nell'esempio soprastante il container è stato avviato come processo in background, tramite l'opzione `--daemon` di `lxc-start`. Potrete interrompere il container attraverso il comando `lxc-stop --name = testlxc`.

Il pacchetto `lxc` contiene uno script di inizializzazione che può avviare automaticamente uno o più contenitori all'avvio dell'host (si basa su `lxc-autostart`, che avvia a sua volta i contenitori individuati attraverso l'opzione `lxc.start.auto` impostata su 1). Il fine-grained access control dell'ordine di avvio dei containers è disponibile grazie alle opzioni `lxc.start.order` e `lxc.group`: di default, lo script di inizializzazione avvia prima i contenitori che sono nel gruppo `onboot` e poi i contenitori che non appartengono a nessuno gruppo. In ogni caso, l'ordine di avvio all'interno di un gruppo viene definito attraverso l'opzione `lxc.start.order`.

[Genericamente il fine-grained access control o controllo a grana fine è un metodo per gestire le variabili di accesso o di non accesso ad una risorsa]

LXC è una soluzione di isolamento abbastanza leggera, pertanto può essere particolarmente adatta all'hosting massivo di servers virtuali. Probabilmente la configurazione di rete con cui vi confronterete sarà più complessa rispetto a quella descritta nel precedente esempio ma le interfacce `tap` e `veth` illustrate nella sopracitata configurazione esemplificativa saranno sufficienti in diversi casi. Potreste anche voler condividere una parte del file system, come ad esempio i subtrees `/usr` e `/lib`, in modo da non dover duplicare il software in comune in diversi contenitori. Le voci `lxc.mount.entry` correlate verranno aggiunte nel file di configurazione dei contenitori. Un interessante effetto collaterale è che i processi occuperanno meno memoria fisica [ossia la RAM], dato che il kernel è in grado di rilevare che i programmi sono stati condivisi. [Il side-effect o effetto collaterale è un effetto non previsto o alterazione tangibile dei valori di una variabile di stato di un'operazione, una funzione o un'espressione, che si aggiunge all'effetto previsto dalla mera invocazione dell'operazione, funzione o espressione in sé.] Il costo marginale di un contenitore aggiuntivo sarà quindi ridotto allo spazio sul disco dedicato ai suoi dati e ad alcuni processi aggiuntivi che devono essere programmati e gestiti dal kernel.

In questa sede non saranno descritte tutte le opzioni disponibili; per maggiori informazioni potrete fare riferimento alle manual pages di `lxc(7)`, `lxc.container.conf(5)`.

12.2.3. Virtualizzazione con KVM

KVM, acronimo di Kernel-based Virtual Machine, è innanzitutto e soprattutto un modulo del kernel che sostiene la maggior parte dell'infrastruttura su cui si regge un virtualizer, ma di per sé non è un virtualizer. L'effettivo controllo della virtualizzazione in sé viene gestito da un applicazione che si basa su QEMU. Di conseguenza non sorprendetevi se in questo paragrafo saranno menzionati i comandi `qemu-*`: si tratta comunque di KVM!

A differenza di altre soluzioni di virtualizzazione, KVM è stato integrato nel kernel Linux sin dai suoi albori. I suoi sviluppatori scelsero di affidarsi ai sets, di istruzioni dei processori, dedicati alla virtualizzazione (Intel-VT o AMD-V), in modo da rendere KVM leggero, elegante e poco dispendioso in termini di risorse. Il rovescio della medaglia è che KVM non funziona su tutti i computer, ma solo su quelli con processori adeguati. Nei computer basati su x86, dovrete verificare se il processore dispone dei sets di istruzioni richiesti, cercando tra i flags elencati in `/proc/cpuinfo` "vmx" o "svm". Grazie a Red Hat, che supporta attivamente il suo sviluppo, KVM è diventato più o meno la virtualizzazione in Linux per antonomasia.

12.2.3.1 Fasi preliminari

Differentemente da noti tools come Virtualbox, KVM non dispone di un'interfaccia-utente per la creazione e la gestione di macchine virtuali. Il pacchetto `qemu-kvm` fornisce solo un eseguibile con lo stesso nome in grado di avviare una macchina virtuale ed anche uno script di inizializzazione che carica i moduli del kernel adeguati.

Fortunatamente, RedHat ha provveduto al supporto di un altro set di strumenti per risolvere questo problema, attraverso lo sviluppo di `libvirt` e degli strumenti correlati al `virtual machine manager`. `libvirt` è una libreria che permette di gestire le macchine virtuali in modo uniforme, indipendentemente dalla tecnologia di virtualizzazione che opera dietro le quinte (e supporta QEMU, KVM, Xen, LXC, OpenVZ, VirtualBox, VMWare e UML). `virtual-manager` è un'interfaccia grafica che utilizza `libvirt` per creare e gestire le macchine virtuali.

Dovrete prima ovviamente installare tutti i pacchetti richiesti attraverso apt-get install qemu-kvm libvirt-bin virtinst virtual-manager virt-viewer. libvirt-bin supporta il demone libvirtd che (potenzialmente da remoto) consente di gestire le macchine virtuali dell'host ed avviare le VMs [le macchine virtuali] richieste durante l'avvio dell'host. Inoltre, il pacchetto supporta virsh, uno strumento da riga di comando, che consente di controllare le macchine virtuali gestite da libvirtd.

Il pacchetto virtinst conferisce virt-install, che consente la creazione di macchine virtuali da riga di comando. Infine, virt-viewer rende possibile la presenza di una console grafica destinata alla macchine virtuali.

12.2.3.2 Configurazione di rete

[*bridge group è il metodo che consente di “raggruppare” più ports in un singolo broadcast domain. Un broadcast domain è una divisione logica di una rete dove tutti i nodi possono comunicare tra loro via broadcast e direttamente al secondo livello ISO/OSI, denominato data link layer. Un broadcast domain quindi non ha la necessità di dover usufruire di un router, né dell'istradamento dei dati (che a sua volta si basa sul terzo livello ISO/OSI denominato network layer), ma l'aspetto negativo è che ogni singolo messaggio viene trasmesso a tutti i nodi.]

Come Xen o LXC, la configurazione di rete più comune consiste nel configurare un bridge in cui verranno integrate le interfacce di rete delle macchine virtuali (per maggiori informazioni andate a leggere il paragrafo 12.2.2.2, "Configurazione di rete" a pagina 330).

In alternativa, la configurazione predefinita di KVM assegna un indirizzo privato alla macchina virtuale (entro l'intervallo 192.168.122.0/24) ed abilita il metodo NAT in modo che la macchina possa accedere alla rete esterna.

Nel proseguo di questo paragrafo, verrà presupposto che è stato configurato un bridge br0 ed un'interfaccia di rete fisica eth0 e che quest'ultima sia stata connessa al bridge.

12.2.3.3 Installazione con virt-install

La creazione di una macchina virtuale è molto simile all'installazione di un normale sistema, tranne per il fatto che tutte le funzionalità della macchina sono descritte attraverso una riga di comando apparentemente senza fine.

In pratica ciò significa che il boot della macchina virtuale verrà avviato dall'unità DVD-ROM virtuale con l'esecuzione di Debian Installer da un'immagine Debian DVD ISO memorizzata nell'host system. La macchina virtuale esporterà la console grafica attraverso il protocollo VNC (troverete delle spiegazioni al riguardo nel paragrafo 9.2.2, “Come utilizzare i Remote Graphical Desktops” a pagina 195) che vi consentirà di controllare l'avanzamento del processo di installazione.

Per prima cosa, qualora intendiate utilizzare una posizione diversa da quella predefinita (/var/lib/libvirt/images/) dovete trasmettere a libvirtd dove avete memorizzato le immagini del disco.

```
root@mirwiz:~# mkdir /srv/kvm
root@mirwiz:~# virsh pool-create-as srv-kvm dir --target /srv/kvm
Pool srv-kvm created
root@mirwiz:~#
```

SUGGERIMENTO
Aggiungete un utente al gruppo libvirt

Tutti gli esempi in questo paragrafo presuppongono che abbiate eseguito i comandi come root. Difatti per controllare il demone libvirt locale, occorre disporre o dell'account root o essere un membro del gruppo libvirt (che non è un'impostazione di default). Pertanto onde evitare di usare i diritti di root, potrete aggiungere il vostro account al gruppo libvirt ed eseguire i vari comandi con la vostra identità utente.

Da qui in poi verrà descritto il processo di installazione della macchina virtuale ed è bene che diate un'occhiata al significato delle opzioni più importanti di `virt-install`. Sarà questo comando a registrare la macchina virtuale ed i suoi parametri in `libvirtd`, nonché ad avviare la macchina virtuale per la prima volta in modo che l'installazione possa procedere.

```
# virt-install --connect qemu:///system
    1
      --virt-type kvm
    2
      --name testkvm
    3
      --ram 1024
    4
      --disk/srv/kvm/testkvm.qcow,format=qcow2,size=10
    5
      --cdrom /srv/isos/debian-8.1.0-amd64-netinst.iso
    6
      --network bridge=br0
    7
      --vnc
    8
      --os-type linux
    9
      --os-variant debianwheezy
```

```
Starting install...
Allocating 'testkvm.qcow'          | 10 GB   00:00
Creating domain...                 | 0 B     00:00
Guest installation complete... restarting guest.
```

1 L'opzione `--connect` consente di specificare "l'hypervisor" da utilizzare. Il modello di questa opzione prevede che venga indicata sotto forma di URL sia la tecnologia di virtualizzazione (`xen://`, `qemu://`, `lxc://`, `openvz://`, `vbox://`, ecc.), sia la macchina host che ospiterà la VM (quest'ultima informazione viene omessa se l'host è la macchina locale). Inoltre, nel caso di QEMU/KVM, ogni utente può gestire le macchine virtuali ponendo dei limiti sui loro permessi o titolarità di diritti attraverso il percorso URL che consente di distinguere le macchine "di sistema" (`/system`) da quelle di "sessione" (`/session`).

2 Dato che KVM viene gestito con le stesse modalità di QEMU, l'opzione `--virt-type kvm` consente di specificare che, nonostante l'URL indichi QEMU, in realtà si tratta di KVM.

3 L'opzione `--name` definisce l'identificatore unico (ossia il nome) assegnato alla macchina virtuale.

4 L'opzione `--ram` definisce la quantità di RAM (in MB) da assegnare alla macchina virtuale.

5 L'opzione `--disk` indica la posizione del file immagine che di fatto rappresenta il disco rigido della macchina virtuale; se il file non esiste, viene creato ad hoc con una dimensione (in GB) pari a quella specificata nel parametro `size`. Il parametro `format` consente di scegliere, tra diverse alternative, il formato di archiviazione del disco virtuale. Il formato predefinito (`raw`) è un file che soddisfa pienamente sia le dimensioni del disco, sia i suoi contenuti. Il formato scelto nell'esempio è tecnologicamente più avanzato, specificatamente destinato a QEMU e permette di associare alla macchina virtuale inizialmente un file di piccole dimensioni, che cresce in proporzione all'effettiva necessità di spazio della macchina virtuale.

6 L'opzione `--cdrom` indica la posizione dell'immagine ISO del disco ottico da utilizzare per avviare l'installazione. Potrete utilizzare come path (percorso): il percorso locale di un file ISO locale; l'URL da cui è possibile recuperare un file ISO disponibile; oppure il percorso locale del device file dell'unità fisica (CD-ROM) (ad es. `/dev/cdrom`).

7 L'opzione --network consente di stabilire come la scheda di rete virtuale dovrà interagire con la configurazione di rete dell'host. Il funzionamento predefinito (esplicitamente “richiesto” nell'esempio) prevede l'integrazione della scheda in un qualsiasi bridge di rete preesistente. In assenza di un bridge, la macchina virtuale avrà accesso alla rete fisica tramite NAT e quindi le verrà assegnato uno degli indirizzi di un intervallo private subnet range (192.168.122.0/24).

8 --vnc stabilisce che la console grafica venga sostenuta da VNC. Per impostazione predefinita, il funzionamento del server VNC correlato prevede soltanto la ricezione dell'interfaccia locale [localhost]; se il client VNC viene messo in esecuzione su un altro host, dovete configurare un tunnel SSH per stabilire la connessione (andate a leggere al riguardo il paragrafo 9.2.1.3, “Port Forwarding: come creare gli Encrypted Tunnels” pagina 194). In alternativa, potrete utilizzare --vnclisten=0.0.0.0 per rendere il server VNC accessibile da qualsiasi interfaccia, a condizione di configurare il firewall adeguatamente a tale scopo.

9 Le opzioni --os-type e --os-variant consentono di ottimizzare alcuni parametri della macchina virtuale, in base alle caratteristiche note del sistema operativo che menzionerete.

Giunti a questo punto la macchina virtuale viene avviata ed è necessario accedere alla console grafica per procedere con l'installazione. Se l'operazione precedente è stata eseguita da un ambiente desktop grafico, la connessione alla console grafica dovrebbe avviarsi automaticamente. Qualora non fosse così, o qualora stiate procedendo da remoto, potrete aprire la console grafica eseguendo da un computer qualsiasi, con un ambiente grafico qualsiasi, `virt-viewer` (occorre precisare che la password di root dell'host remoto verrà richiesta due volte in quanto l'operazione in sé richiede 2 connessioni SSH):

```
$ virt-viewer --connect qemu+ssh://root@server/system testkvm
root@server's password:
root@server's password:
```

Al termine dell'installazione, la macchina virtuale verrà riavviata per essere pronta all'uso.

12.2.3.4 Gestione delle macchine con virsh

L'installazione è completata e quindi è arrivato il momento di descrivere come gestire le macchine virtuali finalmente a disposizione. Per prima cosa dovete verificare le macchine gestite da libvirtd richiedendone attraverso il seguente comando l'elenco:

```
# virsh -c qemu:///system list --all
Id Name State
-----
- testkvm shut off
```

Poi potrete effettuare un test di prova della macchina virtuale:

```
# virsh -c qemu:///system start testkvm
Domain testkvm started
```

Giunti a questo punto potrete ottenere informazioni sulla connessione della console grafica (il VNC display [port] restituito in risposta alla vostra richiesta potrà essere trasmesso a sua volta come parametro a `vncviewer`):

```
# virsh -c qemu:///system vncdisplay testkvm  
:0
```

Sono disponibili altri sotto-comandi di virsh che includono:

- reboot per richiedere un riavvio;
- shutdown per azionare uno spegnimento “clean” [“pulito”, in gergo informatico indica estensivamente un’operazione consona, magistrale, ortodossa, completa] della macchina virtuale;
- destroy interrompe la macchina bruscamente;
- suspend mette la macchina in pausa;
- resume riattiva la macchina dalla pausa;
- autostart per abilitare (o disabilitare, se utilizzato con l’opzione --disable) l’avvio automatico di una macchina virtuale durante l’avvio dell’host;
- undefine per rimuovere tutte le tracce della macchina virtuale da libvirtd.

Tutti questi sotto-comandi utilizzano un identificatore della macchina virtuale come parametro.

12.3.5 Come installare un sistema RPM su Debian attraverso yum

Se state progettando la vostra macchina virtuale con il fine di eseguire Debian (o uno dei suoi derivati), linizializzazione del sistema può essere avviata con debootstrap, come descritto in precedenza. Se invece state progettando la vostra macchina virtuale in modo che esegua un sistema basato su RPM (come Fedora, CentOS o Scientific Linux), dovete eseguirne la configurazione attraverso lutility yum (disponibile nel pacchetto omonimo).

La procedura in sé prevede luso di un rpm per estrarre una serie di files iniziali, inclusi in particolare i files di configurazione di yum, in modo che possiate richiedere a yum di estrarre il resto dei pacchetti. Ma dato che state interagendo con yum dallesterno di chroot, dovete apportare alcune modifiche temporanee. Nell'esempio seguente, la chroot in questione è /srv/centos.

```
# rootdir="/srv/centos"  
# mkdir -p "$rootdir" /etc/rpm  
# echo "%_dbpath /var/lib/rpm" > /etc/rpm/macros.dbpath  
# wget http://mirror.centos.org/centos/7/os/x86_64/Packages/centos-  
release-7-1.1503.  
-> el7.centos.2.8.x86_64.rpm  
# rpm --nodeps --root "$rootdir" -i centos-  
release-7-1.1503.el7.centos.2.8.x86_64.rpm  
rpm: RPM should not be used directly install RPM packages, use Alien instead!  
rpm: However assuming you know what you are doing...  
warning: centos-release-7-1.1503.el7.centos.2.8.x86_64.rpm: Header V3 RSA/SHA256  
-> Signature, key ID f4a80eb5: NOKEY  
# sed -i -e "s,gpgkey=file:///etc/,gpgkey=file://${rootdir}/etc/,g" $rootdir/  
etc/yum.  
-> repos.d/*.repo  
# yum --assumeyes --installroot $rootdir groupinstall core  
[...]  
# sed -i -e "s,gpgkey=file://${rootdir}/etc/,gpgkey=file:///etc/,g" $rootdir/  
etc/yum.  
-> repos.d/*.repo
```

12.3. Installazione automatizzata

Gli amministratori della Falcot Corp, come qualunque altro amministratore di servizi IT, hanno bisogno di strumenti per installare (o reinstallare) rapidamente e, se possibile, automaticamente le nuove macchine acquisite.

Esistono diverse soluzioni per soddisfare tali necessità. Si va da strumenti generici come SystemImager che si occupano sia della creazione dell'immagine basata su una template machine (una macchina modello), sia del deployment della suddetta immagine nei sistemi target; sino a strumenti standard come debian-installer, compatibili con il preseeding che consiste nella compilazione pregressa di un file di configurazione in modo che già contenga le risposte alle varie domande poste durante il processo di installazione [di debian-installer]. Esiste anche una categoria che si frappone fra le due sopracitate, di cui uno strumento ibrido come FAI (Fully Automatic Installer) ne fa parte; FAI è uno strumento che consente sia l'installazione delle macchine sfruttando il sistema di gestione dei pacchetti, sia altre attività relative al deployment massivo (avvio, partizionamento, configurazione, ecc.) utilizzando la sua stessa infrastruttura.

Ognuna di queste soluzioni presenta vantaggi e svantaggi: SystemImager non dipende da un particolare sistema di gestione di pacchetti e quindi consente di gestire una flotta di macchine che eseguono distribuzioni di Linux diverse fra loro. Inoltre questo strumento offre anche un sistema di aggiornamento che non richiede la reinstallazione [delle macchine della flotta]. D'altronde è necessario che le macchine della flotta non vengano modificate indipendentemente, affinché gli aggiornamenti siano affidabili; in altre parole, l'utente non deve aggiornare il software (dagli strumenti dello stesso software) o addirittura installare altri software. Per di più, gli aggiornamenti di sicurezza, non potranno essere automatizzati, ma dovranno essere eseguiti sempre attraverso l'immagine di riferimento di SystemImager. Infine, questa soluzione richiede che le macchine della flotta abbiano caratteristiche omogenee fra loro onde evitare che l'amministratore debba destreggiarsi con diverse immagini (in pratica un'immagine powerpc non sarà compatibile con una macchina i386 e viceversa!).

L'installazione automatizzata può invece essere maggiormente adeguata alle specificità delle diverse macchine attraverso debian-installer: difatti lo strumento dapprima seleziona il kernel ed il software appropriato dai corrispondenti repositories; dopodiché rileva l'hardware e partiziona l'intero disco per utilizzarne tutto lo spazio disponibile; infine installa il sistema Debian correlato all'architettura della macchina e configura un bootloader. Purtroppo però lo svantaggio è che la versione predefinita di debian-installer, installa solo le versioni "standard" di Debian ossia il sistema predefinito e le "tasks" preselezionate; quindi non potrete installare un sistema personalizzato che includa applicazioni non "pacchettizzate". Per risolvere tali problematiche dovete personalizzare l'installer... Fortunatamente, l'installer in sé è piuttosto modulare ed esistono degli strumenti in grado di automatizzare la maggior parte del processo di personalizzazione, tra cui uno dei più noti è simple-CDD (CDD è l'acronimo di Custom Debian Derivative). In ogni caso anche simple-CDD ha i suoi limiti, in quanto questa soluzione gestisce solo le installazioni iniziali; ciò nonostante non si tratta di una reale problematica dato che gli strumenti APT consentono un efficiente deployment degli aggiornamenti successivi.

A seguire troverete soltanto una descrizione approssimativa di FAI e non di SystemImager (visto che non fa più parte di Debian), allo scopo di approfondire soluzioni più pertinenti in un contesto Debian come debian-installer e simple-CDD.

12.3.1. Fully Automatic Installer (FAI)

Il Fully Automatic Installer è probabilmente una delle soluzioni di deployment automatizzate più datate di Debian. Ecco perché questo strumento viene considerato un modello; però la sua stessa natura flessibile difficilmente compensa le complessità che comporta.

FAI necessita di un server system che conservi le informazioni sul deployment e consenta l'avvio delle macchine dalla rete. Il server system a sua volta richiede il pacchetto fai-server (o fai-quickstart che include tutti gli elementi necessari per una configurazione standard).

FAI impiega metodi specifici per identificare i diversi profili potenzialmente installabili. Difatti invece di duplicare meramente un'installazione modello, FAI da perfetto full-fledged installer [trad. non lett. "installer pienamente in grado di soddisfare le esigenze dell'utenza"] è completamente configurabile attraverso una serie di files e scripts memorizzati sul server; la posizione predefinita /srv/fai/config/ non viene creata automaticamente, pertanto l'amministratore dovrà creare sia la sopraccitata directory, sia tutti i files necessari. Nella maggior parte dei casi potrete personalizzare i suddetti files facendo riferimento a degli esempi disponibili nella documentazione inclusa nel pacchetto fai-doc e più specificatamente nella directory /usr/share/doc/fai-doc/examples/simple/.

Una volta che avrete realizzato e definito i profili, dovete eseguire fai-setup per generare i vari elementi necessari per iniziare un'installazione FAI; sostanzialmente si tratta della preparazione (o aggiornamento) di un sistema minimale (NFS-root) da utilizzare durante l'installazione. In alternativa potrete generare un CD di avvio dedicato attraverso fai-cd.

La realizzazione in sé dei suddetti files di configurazione richiede la conoscenza delle metodologie attraverso cui FAI opera. Un'installazione tipica prevede i seguenti passaggi:

- recupero ed avvio del kernel da rete [il fetching (recupero in ingl.) in realtà è anche una delle tre fasi che il processore, dall'avvio del computer, utilizza per processare l'informazione (Fetch, Decode, Execute) e consiste nel recupero della nuova istruzione da processare];
- mounting del root filesystem da NFS [il sopraccitato nfsroot];
- esecuzione di /usr/sbin/fai che controlla il resto dell'installazione (i seguenti passaggi sono quindi avviati da questo script);
- duplicazione dal server del configuration space [che altri non è che un registro dedicato alla gerarchia delle directories] per renderlo disponibile in /fai/;
- esecuzione fai-class. [Di fatto le classi del configuration space ed i loro nomi corrispondono ai servizi a cui sono dedicati gli script di configurazione] Gli scripts /fai/class/[0-9][0-9]* vengono eseguiti e restituiscono i names of "classess" [i nomi di "classe"] da applicare alla macchina che si sta installando; queste informazioni verranno riutilizzate nei vari passaggi a seguire. Questo è un modo relativamente flessibile per definire quali servizi dovranno essere installati e configurati.
- recupero/fetching delle configuration variables in base alle correlate classi;
- partizionamento dei dischi e formattazione delle partizioni, in base alle informazioni presenti in /fai/disk_config/class;
- mounting delle suddette partizioni;
- installazione del sistema di base;
- preseeding del database Debconf attraverso fai-debconf;
- recupero/fetching dell'elenco dei pacchetti disponibili per APT;
- installazione dei pacchetti elencati in /fai/package_config/class;
- esecuzione degli scripts post-configurazione /fai/scripts/class/[0-9][0-9]*;
- registrazione degli installation logs, unmounting delle partizioni, riavvio.

12.3.2. Preseeding Debian-Installer

In linea teorica il miglior strumento per installare i sistemi Debian dovrebbe essere per forza di cose l'installer ufficiale di Debian. Questo è il motivo per cui debian-installer, fin dalla sua progettazione, è stato concepito per essere utilizzato automaticamente, grazie all'ausilio dell'infrastruttura supportata da debconf.

Ciò consente, da un lato, di limitare il numero di domande poste (e qualora ci fossero domande omesse verranno utilizzate le risposte predefinite), dall'altro di preparare le risposte separatamente in modo che l'installazione non debba essere necessariamente interattiva. Questa funzionalità viene definita **preseeding** [e può essere semplicemente tradotta come preconfigurazione].

ANDANDO OLTRE	[Genericamente un database centralizzato è una raccolta di informazioni in un'unica posizione, differentemente da un database distribuito.] Il preseeding consente di definire una serie di risposte per l'installazione, ma questo metodo è statico visto che le risposte non vengono modificate successivamente. Per ovviare a questa problematica, dato che le macchine già installate potrebbero avere necessità di un aggiornamento, potrete configurare debconf attraverso il file di configurazione /etc/debconf.conf per stabilire l'impiego di sorgenti di dati esterne (come un LDAP directory server o un file remoto reso disponibile da NFS o Samba). Potrete utilizzare diverse sorgenti dati contemporaneamente in modo che si completino a vicenda. Solo il database locale sarà accessibile in modalità di lettura-scrittura, mentre gli altri database remoti generalmente si limitano soltanto alla modalità di lettura. La man page debconf.conf(5) descrive dettagliatamente tutto ciò (per consultarla dovete installare il pacchetto debconf-doc).
Debconf con un database centralizzato	

12.3.2.1 Come utilizzare un Preseeding File

Il programma di installazione può recuperare un preseeding file da posizioni diverse:

- nell'initrd utilizzato per avviare la macchina; in questo caso, il preseeding viene implementato all'inizio dell'installazione e così potrete evitare tutte le domande. Vi basterà chiamare il file preseed.cfg e posizionarlo in initrd root.
- sul supporto media per l'avvio (CD o chiavetta USB); in questo caso il preseeding viene implementato durante il mounting del supporto in questione, cioè subito dopo le domande relative alla lingua e al layout della tastiera. Il parametro boot del preseed/file viene utilizzato per indicare la posizione del preseeding file (ad esempio /cdrom/preseed.cfg se si utilizza un CD-Rom o /hd-media/preseed.cfg nel caso di una chiave USB).
- dalla rete; il preseeding viene implementato dopo la configurazione automatica della rete; il parametro pertinente è preseed/url=http://server/preseed.cfg.

Includere il file di preseeding nell'initrd potrebbe sembrare la soluzione migliore, ma dal punto di vista pratico non è così; infatti tale soluzione viene impiegata molto raramente, a causa della complessità della creazione di un installer initrd. Vengono preferite quindi altre due soluzioni, dal momento che i parametri di boot consentono di preconfigurare le risposte alle prime domande. Per evitare la scocciatura di dovere inserire manualmente i parametri di boot ad ogni installazione, potrete salvarli nella configurazione destinata a isolinux (avvio da CD-Rom) o syslinux (avvio da chiavetta USB).

12.3.2.2 Creazione di un Preseed File

Un preseed file è un file plain text [testo puro] in cui ogni riga contiene una risposta ad una domanda Debconf. Ciascuna riga è suddivisa in 4 campi separati da whitespace (spazi o tabs) come ad esempio di d-i mirror/suite string stable:

- Il primo campo è l'owner della domanda; d-i viene impiegato per le domande pertinenti all'installer oppure per indicare il nome di un pacchetto per soddisfare le domande Debconf utilizzate dai pacchetti Debian.
- Il secondo campo definisce un *identifier* per la domanda.
- Il terzo campo stabilisce il tipo di domanda.
- Infine, il quarto campo contiene il valore della risposta. Occorre precisare che questo campo è separato dal terzo campo da un singolo spazio; quindi se immetterete più spazi verranno considerati parte del quarto campo ossia del valore.

Il modo più semplice per creare un *preseed file* è installare un sistema manualmente. Potrete ottenere tutte le risposte relative a *debian-installer* attraverso *debconf-get-selections --installer*; per ottenere le risposte inerenti ad un pacchetto potrete utilizzare *debconf-get-selections*. Tuttavia, è consigliabile scrivere un file di *preseeding* manualmente da un esempio o da documentazione di riferimento: potrete preconfigurare sole le risposte predefinite “sovrascrivibili” alle domande oppure potrete impiegare il parametro *boot priority=critical* per imporre a Debconf di chiedervi solo le domande critiche e di utilizzare le risposte predefinite con le domande non critiche.

DOCUMENTAZIONE Appendice del manuale di installazione

La guida all'installazione, disponibile online, include documentazione in dettaglio riguardo all'impiego di un *preseed file*. Inoltre troverete anche un esempio dettagliato e commentato di un *preseed file* da impiegare come modello per le proprie necessità locali.

- ♦ <https://www.debian.org/releases/jessie/amd64/apb.html>
- ♦ <https://www.debian.org/releases/jessie/example-preseed.txt>

12.3.2.3 Creazione di un supporto di avvio Media personalizzato

La conoscenza nozionistica riguardo alla mera posizione di un *preseed file* non è tutto, in quanto dovete anche essere in grado di mettere in atto la suddetta procedura ossia, in base al metodo prescelto, dovete essere capaci di personalizzare il supporto di avvio dell'installazione allo scopo di modificare i parametri di avvio e aggiungere il *preseed file*.

12.3.2.3.1 Avvio dalla rete Durante il boot da rete, il server responsabile dell'invio degli elementi di inizializzazione definisce anche i parametri di boot. Pertanto dovete includere le personalizzazioni nella configurazione PXE destinata al boot server, ovvero il file di configurazione */tftpboot/pixelinux.cfg/default*. La configurazione dell'avvio da rete è un prerequisito; troverete maggiori dettagli nella Guida all'Installazione.

- ♦ <https://www.debian.org/releases/jessie/amd64/ch04s05.html>

12.3.2.3.2 Preparazione di una chiave USB di avvio. Dopo aver preparato una chiave USB di avvio (troverete maggiori dettagli nel paragrafo 4.1.2, “Booting (avviamento) da una chiave USB” a pagina 49), dovete mettere in atto solo poche operazioni successive. Si presuma ad esempio che il contenuto della chiave USB sia accessibile in */media/usbdisk/*:

- dovete copiare il *preseed file* in */media/usbdisk/preseed.cfg*

- poi dovete modificare /media/usbdisk/syslinux.cfg per aggiungere i parametri di avvio desiderati (in basso ne viene riportato l'esempio).

Esempio 12.2 File syslinux.cfg e parametri per la preconfigurazione

```
default vmlinuz
append preseed/file=/hd-media/preseed.cfg locale=en_US.UTF-8 keymap=us
language=us
-> country=US vga=788 initrd=initrd.gz --
```

12.3.2.3.3 Creare un'immagine CD-Rom Una chiave USB è un supporto di archiviazione accessibile in modalità di lettura/scrittura, pertanto è facile aggiungere un file e modificare alcuni parametri. Purtroppo ciò non è fattibile nel caso di un CD-Rom in quanto l'operazione è più complessa, dato che occorre rigenerare un'immagine ISO full di Debian. Questa attività viene gestita da debian-cd e sfortunatamente, questo strumento è poco intuitivo nel suo impiego: richiede un mirror Debian locale e l'attenta analisi nonché la comprensione dell'amministratore di tutte le opzioni offerte da /usr/share/debian-cd/CONF.sh; dopodiché dovete invocare make diverse volte. È consigliabile la consultazione di /usr/share/debian-cd/README.

Occorre però precisare che debian-cd procede in modo simile: crea dapprima una directory "image" che include tutti i contenuti del futuro CD-Rom e poi usufruisce di un programma (genisoimage, mkisofs o xorriso) per convertire la sopracitata directory in un file ISO. La directory viene ultimata quando viene espletata la fase make image-trees di debian-cd. A questo punto dovete inserire il preseed file nella directory appropriata (solitamente i parametri definiti dal file di configurazione CONF.sh sono \$TDIR/\$CODENAME/CD1/, \$TDIR e \$CODENAME). Il CD-ROM usa come bootloader isolinux; a seguito della configurazione personalizzata di debian-cd dovete modificare anche il file di configurazione di isolinux per includere i parametri di avvio necessari (il file da modificare è \$TDIR/\$CODENAME/boot1/isolinux/isolinux.cfg). Infine, potrete riprendere il "normale" processo di creazione dell'immagine ISO attraverso make image CD=1 (o make images se dovete gestire diversi CD-Roms).

12.3.3. Simple-CDD: la soluzione all-in-one

Purtroppo un preseed file non è sufficiente a soddisfare tutte le esigenze che potrebbero presentarsi nei vasti deployments. Difatti nonostante sia possibile eseguire alcuni scripts al termine del normale processo di installazione, la scelta dei pacchetti da installare rimane limitatamente flessibile (ovvero possono essere selezionate solamente le "tasks"); inoltre questa modalità consente di installare pacchetti ufficiali di Debian e preclude la possibilità dell'installazione di pacchetti generati in locale.

Ad ogni modo, debian-cd è in grado di integrare i pacchetti esterni e debian-installer può essere personalizzato con l'inclusione di passaggi ad hoc nell'installazione. Combinando queste funzionalità, potrete personalizzare l'installer affinché soddisfi le vostre esigenze; dovreste anche configurare i servizi dei pacchetti prescelti dopo l'unpacking. Ovviamente non si tratta di mere parole, dato che quanto appena descritto viene esattamente implementato da Simple-CDD (incluso nel pacchetto simple-cdd).

Lo scopo di Simple-CDD è consentire a chiunque di creare facilmente una distribuzione derivata da Debian, attraverso la selezione di un subset di pacchetti, preconfigurandoli con Debconf,

aggiungendo alcuni software specifici ed eseguendo scripts personalizzati alla fine del processo di installazione. De facto il suddetto tool ottempera “alla filosofia del sistema operativo universale” che chiunque deve essere in grado di adattare alle proprie esigenze.

12.3.3.1 Creazione dei profili

Simple-CDD consente la creazione dei "profili", che corrispondono al concetto delle "classi" FAI; una macchina può avere diversi profili (stabiliti durante il processo di installazione). Un profilo è definito da un insieme di files `profiles/profile.*`:

- il file `.description` contiene una riga di descrizione del profilo;
- il file `.packages` include un elenco di pacchetti che verranno automaticamente installati se il profilo correlato verrà selezionato .
- il file `.downloads` contiene l'elenco dei pacchetti integrati nel supporto media di installazione, ma che non necessariamente verranno installati.
- Il file `.preseed` contiene le informazioni necessarie per il preseeding delle domande Debconf (sia per l'installer, sia per i pacchetti).
- Il file `.postinst` contiene uno script che viene eseguito sul sistema installato alla fine dell'installazione.
- infine, il file `.conf` permette di modificare diversi parametri di Simple-CDD a seconda dei profili inclusi in un'immagine.

Il profilo predefinito ha un particolare scopo e difatti viene configurato sistematicamente: include lo stretto indispensabile affinché Simple-CDD possa funzionare. Spesso in questo profilo viene modificato il parametro `preseed simple-cdd/profiles`: così potrete evitare la domanda di Simple-CDD riguardo ai profili da installare.

Occorre precisare anche che i comandi dovranno essere invocati dalla parent directory della directory `profiles`.

12.3.3.2 Come configurare ed impiegare build-simple-cdd

BREVE ACCENNO	Un modello dettagliato del file di configurazione	Un modello del file di configurazione di Simple-CDD contenente tutti i parametri utilizzabili si trova nello stesso pacchetto (<code>/usr/share/doc/simple-cdd/example/simple-cdd.conf.detailed.gz</code>). Potrete impiegare il suddetto file d'esempio come punto di partenza per la creazione della vostra configurazione personalizzata.
---------------	---	--

Simple-CDD necessita di diversi parametri per poter implementare le sue funzionalità complete. Tali parametri solitamente vengono immessi in un file di configurazione, di cui `build-simple-cdd` potrà farne uso attraverso la sua stessa opzione `--conf` oppure possono essere immessi direttamente tramite i parametri dedicati di `build-simple-cdd`. In basso viene schematizzato il funzionamento del sopraccitato comando e l'impiego dei suoi stessi parametri:

- Il parametro `profiles` elenca i profili da includere nell'immagine del CD-Rom da generare;

- in base all'elenco dei pacchetti richiesti Simple-CDD scarica i files necessari dal server indicato in server e li raccoglie in un mirror parziale (e li trasmetterà successivamente a debian-cd);
- i pacchetti personalizzati elencati in `local_packages` verranno integrati nel mirror locale;
- `debian-cd` viene poi messo in esecuzione (in una posizione locale predefinita che può essere configurata grazie alla variabile `debian_cd_dir`) con l'elenco dei pacchetti da integrare;
- quando `debian-cd` ultima la preparazione della sua stessa directory, Simple-CDD attua a quest'ultima qualche modifica:
 - colloca i files relativi ai profili nella subdirectory `simple-cdd` (che finirà nel CD-Rom).
 - aggiunge i files elencati nel parametro `all_extras`.
 - modifica i parametri di avvio per attivare il preseeding. Per evitare le prime domande riguardanti la lingua e il paese dovete immettere queste informazioni nelle variabili `language` e `country`.
- dopodiché `debian-cd` genera l'immagine ISO finale.

12.3.3 Creazione dell'immagine ISO

Una volta che avrete ultimato di scrivere il file di configurazione e che avrete definito i profili, dovete invocare `build-simple-cdd --conf simple-cdd.conf`. Dopo pochi minuti, otterrete l'immagine desiderata: `images/debian-8.0-amd64-CD-1.iso`

12.4. Monitoring [Monitoraggio o Supervisione]

Il generico termine di monitoraggio o supervisione è un'accezione che copre diversi aspetti e risponde a diverse questioni difatti: da un lato, il suo significato attiene al mero tracciamento dell'impiego delle risorse di una determinata macchina, al fine di anticipare la saturazione e le conseguenti esigenze di aggiornamento; dall'altro riguarda il mero avviso all'amministratore qualora un servizio diventasse temporaneamente indisponibile o sopraggiungesse un suo malfunzionamento al fine di porvi rimedio nel più breve tempo possibile.

Munin è una valida risposta alla prima questione [il mero monitoraggio], in quanto propone in forma grafica le serie storiche di numerosi parametri (utilizzo della RAM, utilizzo del disco, carico del processore, traffico di rete, carico di Apache/MySQL, ecc.). Nagios risponde invece alla seconda questione, attraverso la verifica regolare dei servizi (del loro funzionamento e della loro disponibilità) e con l'invio di avvisi attraverso canali ad hoc (e-mails, messaggi testuali, ecc.).

Entrambi i programmi sono progettati in modo modulare: quindi sarà per voi relativamente facile creare nuovi plugin per monitorare servizi o parametri specifici.

ALTERNATIVA Zabbix, uno strumento di supervisione interamente integrato	Sebbene Munin e Nagios siano molto diffusi, non sono gli unici software che consentono il monitoraggio, inoltre singolarmente si occupano solo di una specifica area (tavole grafiche o avvisi). Zabbix invece integra entrambe le due attività di monitoraggio ed è in parte configurabile tramite interfaccia web. Negli ultimi anni è notevolmente migliorato, tanto da poter essere considerato una valida alternativa. Sul server di monitoraggio, dovrete installare zabbix-server-pgsql (o zabbix-server-mysql) e preferibilmente anche zabbix-frontend-php per poter disporre di un'interfaccia web. Sugli host da monitorare dovrete installare zabbix-agent in modo che possano trasmettere le informazioni necessarie al server.
	♦ http://www.zabbix.org/

ALTERNATIVA Icinga, un fork di Nagios	A seguito di divergenze di opinioni sullo sviluppo di Nagios (che è controllato da una società), alcuni sviluppatori hanno deciso di creare un fork di Nagios e di chiamarlo Icinga. Icinga è ancora compatibile - per il momento - con le configurazioni ed i plugins di Nagios, ma dispone di ulteriori funzionalità.
	♦ http://www.icinga.org/

12.4.1. Configurazione di Munin

Lo scopo di Munin è di consentire la supervisione di diverse macchine; pertanto impiega un'architettura client/server. Il central host - il grapher - raccoglie i dati esportati dagli altri host da supervisionare e ne elabora le serie storiche in grafici.

12.4.1.1 Configurazione degli host da monitorare

Il primo passo è installare il pacchetto munin-node. Questo pacchetto installa un demone che si pone in ricezione del port 4949 e ne restituisce i dati raccolti da tutti i plugin attivi. Ciascun plugin è un semplice programma in grado di restituire una descrizione dei dati raccolti tra cui l'ultimo valore rilevato. I plugins vengono memorizzati in /usr/share/munin/plugins/, mentre quelli effettivamente in uso vengono memorizzati in /etc/munin/plugins/.

L'installazione iniziale del pacchetto preconfigura un elenco di plugin attivi in base al software disponibile e alla configurazione corrente della macchina. Tuttavia l'autoconfigurazione dipende da una funzionalità integrata di cui ciascun plugin deve disporre ed è quindi preferibile una sua revisione e personalizzazione manuale. Dovreste consultare Plugin Gallery [<http://gallery.munin-monitoring.org>] anche se non tutti i plugins hanno una documentazione completa. In ogni caso tutti i plugin sono scripts, spesso relativamente semplici che includono commenti esplicativi. Potrete accertarvi delle funzionalità di ciascun plugin consultando /etc/munin/plugins/ ed eventualmente potrete rimuovere i plugins non necessari. Inoltre attraverso un symbolic link potrete attivare il plugin che desiderate e che si trova in /usr/share/munin/plugins/ ossia attraverso il comando ln -sf /usr/share/munin/plugins/plugins /etc/munin/plugins/. Occorre precisare che se il nome di un plugin termina con il carattere di underscore ossia il trattino basso (_) significa che il suddetto plugin richiede un parametro. Pertanto dovete specificare tale parametro nel collegamento simbolico che creerete; ad esempio il plugin if_necessita che venga specificata l'interfaccia da monitorare, quindi quando creerete il collegamento simbolico dovete immettere if_eth0 per monitorare il traffico sull'interfaccia di rete eth0.

Quando avrete installato tutti i plugins dovete aggiornare la configurazione del demone per definire il controllo accessi per la raccolta dati. Potrete gestire tutto ciò attraverso le direttive `allow in /etc/munin/munin-node.conf`. Per impostazione predefinita troverete `allow ^127\.0\.0\.1$` che consente solo l'accesso all'host locale. L'amministratore dovrà quindi aggiungere una riga simile contenente l'indirizzo IP della macchina che assumerà il ruolo di grapher e poi riavviare il demone con il servizio `munin-node restart`.

ANDANDO OLTRE

Come creare i plugin locali

Munin dispone di un'ampia documentazione in merito a come i plugins dovrebbero funzionare e come andrebbero sviluppati.

♦ <http://munin-monitoring.org/wiki/plugins>

Per testare il plugin, dovete metterlo in esecuzione nelle stesse condizioni di quelle che si verificherebbero se fosse messo in esecuzione direttamente da `munin-node`; potrete simulare il suddetto contesto eseguendo il plugin attraverso `munin-run` autenticati come utente root. Se definirete un secondo parametro (come ad esempio `config`) quest'ultimo verrà trasmesso al plugin ed eseguito come se fosse un suo parametro.

Se invocherete il plugin con il parametro `config` vi restituirà un insieme di campi che lo descrivono, ad esempio:

```
$ sudo munin-run load config
graph_title Load average
graph_args --base 1000 -l 0
graph_vlabel load
graph_scale no
graph_category system
load.label load
graph_info The load average of the machine describes how
-> many processes are in the run-queue (scheduled to run
-> "immediately").
load.info 5 minute load average
```

I diversi campi sono descritti in “Plugin reference” che fa parte di “Munin guide”.

♦ <http://munin.readthedocs.org/en/latest/reference/plugin.html>

Quando il plugin viene invocato senza parametri, restituisce semplicemente gli ultimi valori rilevati; ad esempio l'esecuzione di `sudo munin-run load` potrebbe restituire `load.value 0.12`.

Infine, quando il plugin viene invocato con `autoconf` come parametro, potrebbe restituire “yes” (con exit status 0 o valore di uscita uguale a 0) o “no” (con exit status 1 o valore di uscita uguale a 1) a seconda se il plugin deve essere abilitato sulla macchina.

12.4.1.2 Configurazione del Grapher

Viene definita “grapher” la macchina che raccoglierà i dati e genererà i grafici corrispondenti. Il software necessario è incluso nel pacchetto `munin`. La configurazione predefinita del pacchetto avvia `munin-cron` ogni 5 minuti, che a sua volta: raccoglie i dati provenienti da tutte le macchine elencate in `/etc/munin/munin.conf` (solo l'host locale viene incluso nell'elenco per impostazione predefinita); memorizza i logs sotto forma di file RRD (Round Robin Database è un formato di file adatto per conservare i dati che variano nel corso tempo) in `/var/lib/munin/`; genera una pagina HTML con tavole grafiche in `/var/cache/munin/www/`.

Dovrete elencare tutte le macchine da monitorare nel file di configurazione `/etc/munin/munin.conf`. Ogni macchina viene definita sotto forma di una sezione completa con un nome corrispondente alla macchina in sé e che include infine la voce `address` con l'indirizzo IP della macchina da supervisionare.

```
[ftp.falcot.com]
address 192.168.0.12
use_node_name yes
```

Le sezioni possono essere più complesse e descrivere ulteriori grafici derivanti dalla combinazione dei dati provenienti da più macchine. Potrete usare come modelli per le vostre personalizzazioni gli esempi inclusi nel file di configurazione.

Infine, l'ultimo passaggio consiste nel pubblicare le pagine generate; per renderlo possibile dovete configurare il server web in modo che i contenuti di `/var/cache/munin/www/` siano disponibili tramite website. Ovviamente in genere l'accesso al suddetto website è limitato da un sistema di autenticazione o da un controllo accessi basato su indirizzi IP. Per maggiori dettagli andate a leggere il paragrafo 11.2, "Server Web (HTTP)" a pagina 268.

12.4.2. Configurazione di Nagios

A differenza di Munin, Nagios non richiede di installare nulla sulle macchine da monitorare; infatti Nagios, nella maggior parte dei casi, viene utilizzato semplicemente per verificare la disponibilità dei servizi di rete. Ad esempio, Nagios può essere impiegato per connettersi e testare un server Web, per poi verificare i tempi di caricamento di una determinata pagina Web.

12.4.2.1 Installazione

Per poter configurare Nagios dovete innanzitutto installare i pacchetti `nagios3`, `nagios-plugins` e `nagios3-doc`. L'installazione dei sopracitati pacchetti configura l'interfaccia web di Nagios e crea un primo utente `nagiosadmin` (per cui sarà chiesta una password). Potrete aggiungere altri utenti inserendoli nel file `/etc/nagios3/htpasswd.users` usando il comando Apache `htpasswd`. Se `debconf` non pone richieste durante l'installazione, potrete eseguire `dpkg-reconfigure nagios3-cgi` per impostare la password per l'utente `nagiosadmin`.

Collegandovi su `http://server/nagios3/` potrete consultare l'interfaccia web e constatare che Nagios sta già monitorando alcuni parametri della macchina su cui è in esecuzione. Tuttavia, alcune funzionalità interattive come l'aggiunta di commenti su un host non sono attive. Per impostazione predefinita e ragioni di sicurezza, Nagios è difatti configurato in modo molto restrittivo e pertanto le suddette funzionalità sono disabilitate.

La documentazione `/usr/share/doc/nagios3/README.Debian` chiarisce al riguardo che occorre modificare `/etc/nagios3/nagios.cfg` ed impostare il parametro `check_external_commands` a "1". Di conseguenza dovrete cambiare anche i permessi di scrittura della directory impiegata da Nagios con i seguenti comandi:

```
# service nagios3 stop
[...]
# dpkg-statoverride --update --add nagios www-data 2710 /var/lib/nagios3/rw
# dpkg-statoverride --update --add nagios nagios 751 /var/lib/nagios3
# service nagios3 start
[...]
```

12.4.2.2 Configurazione

L'interfaccia web di Nagios è relativamente gradevole alla vista, ma non consente alcuna configurazione, né tantomeno l'aggiunta di hosts e servizi da monitorare. Infatti la configurazione di questo software viene gestita attraverso una serie di files definiti nel file di configurazione centralizzato [central configuration file] /etc/nagios3/nagios.cfg.

Prima di illustrare i suddetti files, dovete prendere confidenza con i concetti di Nagios. La configurazione elenca un insieme di oggetti dei seguenti tipi:

- host - è una macchina di rete che si desidera monitorare;
- hostgroup - è un insieme di hosts di cui si intendono monitorare o calcolare gli elementi comuni di configurazione;
- service - è un elemento da verificare che riguarda un host o un host group. Di solito, attraverso questo oggetto viene controllato il funzionamento dei "servizi" di rete, ma può servire anche per riscontrare il livello di accettabilità dei parametri (come ad esempio lo spazio libero sul disco o il carico della CPU);
- servicegroup - è un insieme di servizi da monitorare;
- contact - è una persona che può ricevere avvisi;
- contactgroup - è un insieme di contatti che possono ricevere le notifiche;
- timeperiod - è una fascia oraria in cui devono essere verificati determinati servizi;
- command - è una riga di comando invocata per testare un dato servizio.

Ogni oggetto, in base al tipo correlato, ha un numero di proprietà che possono essere personalizzate. Una loro elencazione completa non è fattibile, ma ciò non ha rilievo dato che le relazioni tra gli oggetti sono le proprietà più importanti.

Un servizio (service) utilizza un comando (command) per verificare lo stato di una funzionalità su un host (o su un hostgroup) durante un determinato arco temporale (timeperiod). In caso di anomalie, Nagios invia un avviso a tutti i membri del contactgroup associato al servizio guasto. Ogni membro viene avvisato attraverso il canale definito nel suo oggetto contact corrispondente.

Il sistema di ereditarietà tra oggetti semplifica la condivisione di un insieme di proprietà tra oggetti diversi in modo che non debba essere duplicata l'informazione. Inoltre, la configurazione iniziale prevede una serie di oggetti standard; nella maggior parte dei casi, è sufficiente definire i nuovi host, servizi e contatti che derivano da oggetti generici predefiniti. I files inclusi in /etc/nagios3/conf.d/ sono una buona fonte di informazioni riguardo al loro funzionamento.

A seguire un esempio della configurazione utilizzata dagli amministratori della Falcot Corp:

Esempio 12.3 Il file /etc/nagios3/conf.d/falcot.cfg

```
define contact{
    name                      generic-contact
    service_notification_period 24x7
    host_notification_period    24x7
    service_notification_options w,u,c,r
    host_notification_options   d,u,r
    service_notification_commands notify-service-by-email
    host_notification_commands  notify-host-by-email
    register                  0 ; Template only
}

define contact{
    use          generic-contact
    contact_name rhertzog
    alias        Raphael Herzog
    email        hertzog@debian.org
}

define contact{
    use          generic-contact
    contact_name rmas
    alias        Roland Mas
    email        lolando@debian.org
}

define contactgroup{
    contactgroup_name   falcot-admins
    alias              Falcot Administrators
    members            rhertzog, rmas
}

define host{
    use generic-host ; Name of host template to use
    host_name www-host
    alias www.falcot.com
    address 192.168.0.5
    contact_groups falcot-admins
    hostgroups debian-servers,ssh-servers
}

define host{
    use generic-host ; Name of host template to use
    host_name ftp-host
    alias ftp.falcot.com
    address 192.168.0.6
    contact_groups falcot-admins
    hostgroups debian-servers,ssh-servers
}
```

```

# 'check_ftp' command with custom parameters
define command{
}
command_name check_ftp2
command_line /usr/lib/nagios/plugins/check_ftp -H $HOSTADDRESS$ -w 20 -c
-> 30-t35
}

# Generic Falcot service
define service{
    name          falcot-service
    use           generic-service
    contact_groups falcot-admins
    register      0
}

# Services to check on www-host
define service{
    use          falcot-service
    host_name   www-host
    service_description HTTP
    check_command check_http
}
define service{
    use          falcot-service
    host_name   www-host
    service_description HTTPS
    check_command check_https
}
define service{
    use          falcot-service
    host_name   www-host
    service_description SMTP
    check_command check_smtp
}
# Services to check on ftp-host
define service{
    use          falcot-service
    host_name   ftp-host
    service_description FTP
    check_command check_ftp2
}

```

Il file di configurazione preso in esame descrive due host da monitorare. Il primo host è il web server che viene monitorato attraverso i ports HTTP (80) e secure-HTTP (443). Nagios controlla anche se un server SMTP è accessibile attraverso il port 25. Il secondo host è il server FTP di cui vengono controllati i tempi di risposta (entro i 20 secondi). Se viene superato l'arco temporale definito il suddetto server è in ritardo (delay), pertanto viene generato un warning; oltre i 30 secondi, l'avviso viene considerato critico. L'interfaccia web di Nagios evidenzia inoltre che il servizio SSH è

monitorato: ciò è dovuto al fatto che gli hosts in questione appartengono all'hostgroup ssh-servers. Il servizio standard corrispondente è definito in /etc/nagios3/conf.d/services_nagios2.cfg.

Nell'esempio soprastante potrete notare l'impiego dell'ereditarietà: un oggetto può essere reso erede di un altro oggetto attraverso la proprietà "use parent-name". Un parent object (oggetto padre o genitore) deve essere identificabile e di conseguenza deve possedere la proprietà "name identifier". Se un parent object non è un oggetto reale, ma è destinato solo a fungere da "genitore", dovrete associarigli la proprietà "register 0" in modo che Nagios sia informato di non prenderlo in considerazione e pertanto di ignorare l'eventuale assenza di alcuni parametri normalmente richiesti.

DOCUMENTAZIONE

Elenco delle proprietà degli oggetti

Per maggiori informazioni in merito alla configurazione di Nagios, dovraste consultare la documentazione inclusa nel pacchetto nagios3-doc. Sarà per voi direttamente accessibile dall'interfaccia web per mezzo del link "Documentation" nell'angolo in alto a sinistra. Inoltre troverete un elenco esaustivo dei diversi tipi di oggetto e di tutte le proprietà che possono essere loro assegnate. Viene anche spiegato come creare nuovi plugin.

ANDANDO OLTRE

Test a distanza con NRPE

Molti plugin Nagios permettono di controllare lo stato di alcuni parametri locali di una macchina; se si desidera eseguire questi controlli su più macchine che fanno parte di un'installazione centralizzata che li raggruppa, dovrete installare con correlato deployment il plugin NRPE (Nagios Remote Plugin Executor). Di conseguenza dovrete installare nagios-nrpe-plugin sul server Nagios e nagios-nrpe-server sulle macchine su cui desiderate eseguire i test locali. Quest'ultimo pacchetto è configurabile tramite il file /etc/nagios/nrpe.cfg. Il file in questione dovrebbe elencare i test che possono essere avviati da remoto e gli indirizzi IP delle macchine autorizzate ad implementarli. Per abilitare i test remoti da Nagios dovrete semplicemente aggiungere i servizi corrispondenti utilizzando il nuovo comando check_nrpe.



Parole chiave
Workstation
Graphical desktop
Office work
X.org

