Diskless Linux Mini Howto

di Robert Nemkin `buci@math.klte.hu`. Tradotto da Salvatore Meschini `smeschini@ermes.it` v0.0.3, 12 Settembre 1996. Tradotto il 30 agosto 1998.

Questo documento descrive come impostare un sistema Linux privo di unità disco ed i diritti d'autore appartengono a Robert Nemkin. La polizza del copyright è GPL. L'autore ringrazia Bela Kis <bkis@cartan.math.klte.hu> per aver tradotto in inglese questo documento.

Indice

1 Modifiche

- v0.0.3 12 Settembre 1996: Correzione di alcuni errori

2 Come impostare un Linux box senza avere unità a disco

Questo documento tratta l'impostazione di una stazione Linux priva di dischi ("diskless"). Ciò può essere necessario per eseguire Linux su PC che non ha né dischi rigidi né floppy drive. Se si opera in una rete con un altro sistema Unix fornito di bootp, tftp, un server NFS, ed è disponibile un programmatore di eprom allora è possibile configurare ed usare Linux senza avere drive.

3 Altra documentazione

- NFS-root Mini Howto
- Linux NET-2/3-HOWTO a cura di Terry Dawson, `94004531@postoffice.csu.edu.au`
- `/usr/src/linux/README` configurazione e compilazione di nuovi kernel

4 Hardware

Tutto ciò che è descritto nel documento è stato provato con la configurazione seguente:

- Sun-OS 4.1.3 come boot server
- Slackware 2.3 + Linux 1.2.8 + wd 8013 ethercard.
- Una LAN Ethernet funzionante

5 Idee basilari

L'idea fondamentale è la seguente: il PC prenderà il suo indirizzo IP dal boot server attraverso il protocollo bootp. È necessario usare 0.0.0.0 come indirizzo IP iniziale ed il kernel del server mediante il protocollo tftp.[1]

Seguite i passi indicati:

5.1 Impostazione del PC

Prendete il pacchetto nfsboot (il pacchetto è disponibile su diversi mirror, nella directory `/pub/Linux/system/Linux-boot`. Nfsboot contiene un'immagine booteprom della scheda wd8013 che può essere programmata direttamente nella EPROM.

Ci sono anche metodi alternativi per preparare il PC:

- Se la vostra macchina ha una qualche unità disco, allora è possibile usare un programmino DOS oppure
- l'immagine binaria per floppy contenuta nel pacchetto precedentemente indicato.

Se preferite adottare la seconda soluzione allora dovete scrivere l'immagine su un floppy utilizzando il comando dd

Queste immagini contengono un client bootp ed uno tftp. Dovete preparare un Linux kernel che supporti l'opzione nfs-root.

[1] La fase di avvio fatta attraverso dei segmenti (via router) non è cosa da poco, dunque conviene mettere sia il server che i terminali senza dischi nello stesso segmento LAN oppure configurare un indirizzo di sostegno UDP nel router verso l'indirizzo del server. Fate riferimento al manuale del vostro router per ottenere ulteriori informazioni.

- Se state usando l'ultimo kernel stabile, linux-1.2.13, allora dovete applicare il patch al kernel sfruttando il file incluso nel pacchetto nfsboot [2]
- Se decidete di usare l'ultimo kernel della serie linux-1.3.x, allora sarà necessario configurare l'opzione nfs-root.

Potete scegliere se configurare o meno il supporto per il block device (floppy o hard disk), ma dovete configurare il supporto per tcp/ip, per le schede ethernet e per il filesystem nfs. Poi ricompilate il kernel come fate di solito.

5.2 Impostazione di bootpd sul server

Bootpd può essere reperito nel pacchetto `bootpd-2.4.tar.gz` (disponibile nella directory `/pub/Linux/system/Network/boot.net` del vostro Linux mirror preferito). Scaricate il pacchetto, compilatelo ed installatelo. Se il sistema Unix presente sul server è Slackware Linux allora potete saltare questo passo poiché la distribuzione contiene un bootpd. Il demone può essere eseguito direttamente digitando il comando

```
bootpd -s
```

oppure usando inetd. In questo caso avete bisogno di editare:

- /etc/inetd.conf per rimuovere il simbolo hash (#) dall'inizio di queste righe:

```
# tftp   dgram  udp   wait   root   /usr/sbin/in.tftpd    tftpd /export
# bootps dgram  udp   wait   root   /usr/sbin/in.bootpd   bootpd
```

- inserite o decommentate le seguenti due linee in /etc/services:

```
bootps        67/tcp        # BOOTP server
tftp          69/udp        # TFTP server
```

- riavviate inetd con

```
kill -HUP <id del processo inetd>.
```

5.3 Configurazione di bootpd sul server.

Per prima cosa, bootpd ha un file di configurazione chiamato bootpdtab che di solito si trova in /etc. Bisogna modificarlo inserendo gli indirizzi IP del vostro gateway, del server dns e gli indirizzi ethernet della macchina sprovvista di unità a disco. Un esempio del file /etc/bootptab:

```
global.prof:\
        :sm=255.255.255.0:\
        :ds=192.168.1.5:\
        :gw=192.168.1.19:\
        :ht=ethernet:\
        :bf=linux:
macchina1:hd=/export/root/machine1:tc=global.prof:ha=0000c0863d7a:ip=192.168.1.140:
macchina2:hd=/export/root/machine2:tc=global.prof:ha=0800110244e1:ip=192.168.1.141:
macchina3:hd=/export/root/machine3:tc=global.prof:ha=0800110244de:ip=192.168.1.142:
```

global.prof è un modello generale per le entrate host, dove

- il campo sm contiene il subnet mask

[2] Fate riferimento a patch(1)

- il campo ds contiene l'indirizzo del Domain Name Server
- il campo gw contiene l'indirizzo del gateway predefinito
- il campo ht contiene il tipo di hardware LAN
- il campo bf contiene il nome del file di avvio

Inoltre ogni macchina deve avere una linea:

- il primo campo contiene il nome dell'host,
- il campo hd contiene la directory del file di avvio,
- il modello globale può essere incluso con il campo tc,
- il campo ha contiene l'indirizzo hardware della scheda ethernet,
- il campo ip contiene l'indirizzo IP assegnato.

5.4 Capire tftp

TFTP (`Trivial File Transfer Protocol`) è un protocollo per il trasferimento di file, come ftp, ma molto più semplice da codificare nelle EPROM. TFTP può essere usato in due modi:

- tftp ingenuo: significa che il client può accedere all'intero file system della vostra macchina. È semplice ma rappresenta un enorme buco nella sicurezza del sistema (chiunque può prelevare il vostro file delle password con tftp).
- tftp sicuro: il server tftp usa una chiamata di sistema di tipo chroot.2 per cambiare la propria directory root. Qualunque cosa all'esterno della nuova directory root sarà completamente inaccessibile. Poiché la directory chroot diventa la nuova directory root, il parametro hd presente nel file bootpdtab deve riflettere la nuova situazione. Per esempio: quando si usa il tftp ingenuo, il campo hd contiene il percorso completo alla directory di boot: `/export/root/machine1`. Quando invece si usa il tftp sicuro con /export come directory di root, /export diviene / ed il campo hd deve essere `/root/macchina1`.

Quasi tutte le implementazioni di Unix contengono il server tftp, probabilmente non ci sarà bisogno di installarne uno.

5.5 Impostazione di una configurazione minimale di Linux su un server remoto.

Una configurazione minimale dovrebbe contenere i pacchetti a, ap, n ed x della distribuzione Slackware. Installare altri pacchetti non è un errore, comunque i pacchetti appena menzionati sono sufficienti ai fini di creare un terminale X senza unità disco. Per l'installazione c'è bisogno di un sistema Linux funzionante. Trovate dello spazio libero sul disco della macchina remota ed esportatelo come lettura-scrittura. Montate la directory esportata da qualche parte (es. /mnt) sul file system del terminale Linux. Avviate il setup di Linux e cambiate l'opzione root nel setup da / a /mnt. Poi installate i pacchetti che ritenete necessari come fate di solito. Se volete usare al massimo un terminale Linux allora non dovete apportare nessun altro cambiamento. Se invece pensate di aver bisogno di più macchine senza unità disco, allora il setup tradizionale non funzionerà perché alcuni file e directory devono risultare private per la macchina. Il problema può essere risolto spostando la directory /usr (non contiene dati privati) e creando una sotto-directory separata per ogni terminale. Per esempio, se /export/linux/macchina1 fosse montata su /mnt allora la struttura delle directory dopo il setup iniziale risulterebbe simile a questa:

```
/export/linux/macchina1/bin
/export/linux/macchina1/sbin
/export/linux/macchina1/lib
/export/linux/macchina1/etc
/export/linux/macchina1/var
/export/linux/macchina1/usr
```

Dopo i cambiamenti la struttura diventerà:

```
/export/linux/macchina1/bin
/export/linux/macchina1/sbin
/export/linux/macchina1/lib
/export/linux/macchina1/etc
/export/linux/macchina1/var
/export/linux/usr
```

Ora dovete creare le sotto-directory per le altre macchine. Da adesso in poi si assuma che i sistemi senza dischi siano chiamati macchina1, macchina2, macchina3, ecc.; In seguito sarà possibile usare il seguente script bash per impostare le altre directory

```
cd /export/linux
for x in macchina2 macchina3 ; do
        mkdir $x; cd $x
        (cd ../macchina1; tar cf - *) | tar xvf -
done
```

Poi create il seguente export:

- /export/linux/usr sola lettura per tutti.
- /export/linux/macchina1 solo alla macchina1 con diritti rw,root.
- /export/linux/macchina2 solo alla macchina2 con diritti rw,root.
- /export/linux/macchina3 solo alla macchina3 con diritti rw,root.

nel modo seguente[3]:

```
# Questo file è /etc/export
# per terminali remoti Linux X (by Buci)
# questa linea va indicata una sola volta
/export/root/usr             -access=linuxnet
# queste linee vanno indicate una volta per ogni host
/export/root/machine1       rw=machine1,root=machine1
/export/root/machine2       rw=machine2,root=machine2
/export/root/machine3       rw=machine3,root=machine3
```

Non dimenticate di eseguire exportfs -a.

5.6 Configurazione del server tftp

È giunto il momento di configurare il server tftp. Se non avete bisogno del tftp sicuro allora tutto è abbastanza facile ed i vostri client possono essere avviati dalla directory /export.

[3]il formato di questo esempio è conforme alla sintassi del SunOs 4.1.3 per i file exports

Se preferite usare il tftp sicuro allora dovete creare una struttura delle directory /export/linux completa sotto /tftpboot (con un solo kernel reale e links simbolici per le altre macchine), oppure lasciate che /export sia la directory di avvio del tftpd sicuro. Se avete una directory tftpboot separata allora ci sarà bisogno solo della struttura delle directory originale con un solo kernel e links simbolici per gli altri terminali. Potete ottenere questo setup digitando i comandi seguenti:

```
mkdir -p /tftpboot/export/linux/macchina1
cd /tftpboot/export/linux/macchina1
cp /export/linux/macchina1/<nome del kernel> .
```

In seguito digitate:

```
mkdir -p /tftpboot/export/linux/macchina2
cd ../macchina2
ln -s ../macchina2/<nome del kernel>
```

5.7 Parte finale

Infine dovete inserire

```
/sbin/mount nfs_server:/export/linux/usr /usr
```

come prima linea di

```
/export/linux/<macchinax>/etc/rc.d/rc.S
```

dove <macchinax> sta per macchina1, macchina2, ecc.

6 Requisiti relativi alla memoria e allo spazio su disco; velocità

Io ho provato questo metodo solo con la Slackware 2.3; per altre distribuzioni/versioni i numeri seguenti possono cambiare.

- Spazio su disco: 28MB + 6.5MB/per ogni macchina
- RAM: Io uso X con 8 MB. Per sistemi con solo 4MB suppongo sia necessario dello spazio swap, createlo – separatamente per ogni macchina – in /tmp. Non dimenticate di lanciare il comando mkswap.
- Velocità: non ho avuto problemi con un 486 DX2/66 con 8 mega di RAM.

7 Problemi

- Mi sono imbattuto in uno strano errore: nella sotto-directory /dev SunOS ha rovinato i dispositivi, perciò ho dovuto rieseguire MAKEDEV montando le sotto-directory su un sistema Linux dotato di unità disco (la causa del problema sta nelle differenze tra il nfs (network filesystem) di Linux e quello di SunOS: tutti e due usano 32 bit per indicare il numero Major e Minor del dispositivo, ma Linux usa campi a 16 bit per entrambi, SunOs usa campi a 14 bit per la parte Major e a 18 bit per quella Minor.).
- Quando il sistema Linux senza unità disco viene avviato, esiste un unico instradamento incluso nella tabella di instradamento al server tftp, così avete bisogno di impostare correttamente le tabelle di instradamento. Potete contare su due possibili scelte:
  - configurare a mano ogni rc.S per ogni macchina
  - usare un client bootp e scrivere uno script di configurazione generalizzato

8 Errori e possibili ulteriori espansioni di questo documento

- Elenco corretto degli altri documenti disponibili
- SunOs è basato su BSD. Inclusione della configurazione del server basato su SVR4 (cioè Solaris)
- Nonostante Linux sia abbastanza simile a SunOs come server bootp/tftp, un esempio di server basato su Linux potrebbe essere utile.
- Aggiornamento di questo documento al pacchetto etherboot corrente.
- Elenco delle differenze tra il network filesystem del kernel 1.2.13 e quello del più recente 1.3.x che contiene il patch nfs-root.
- Necessità di provare altre schede ethernet oltre alla wd8013
- Aggiunta delle informazioni di configurazione per bootpc, un client bootp per Linux utile ad impostare le tabelle di instradamento corrette.
- Refusi ed altri errori: per favore segnalateli a `buci@math.klte.hu`. Grazie.