

Corso formazione ReteIDRA

Savona 23/2 – 30/4 2010
corso avanzato

Alessandro Dentella



Virtualizzazione con netkit



Savona -2010

Labs di oggi

- Oggi usiamo netkit per impraticirci con gli strumenti:
 - Ping
 - Tcpdump
 - Ifconfig
 - Route
 - Arp
 - DHCP

vm

- ogni macchina virtuale “parte” da un filesystem contenuto in un file singolo
- può essere reso scrivibile tramite l'utilizzo della tecnica Copy-on-write (cow) che scrive solo le porzioni di file che cambiano. Il sistema che ne deriva usa poco spazio ma permette di creare tante linux box indipendenti

Vm (segue)

- ogni macchina virtuale richiede poche risorse: è quindi possibile eseguire svariate macchine in contemporanea
- ogni macchina ha una sua interfaccia di rete (o più)
- Le interfacce di rete possono essere connesse fra di loro tramite degli switch virtuali
- E` possibile decidere se devono vedersi fra di loro o meno



netkit

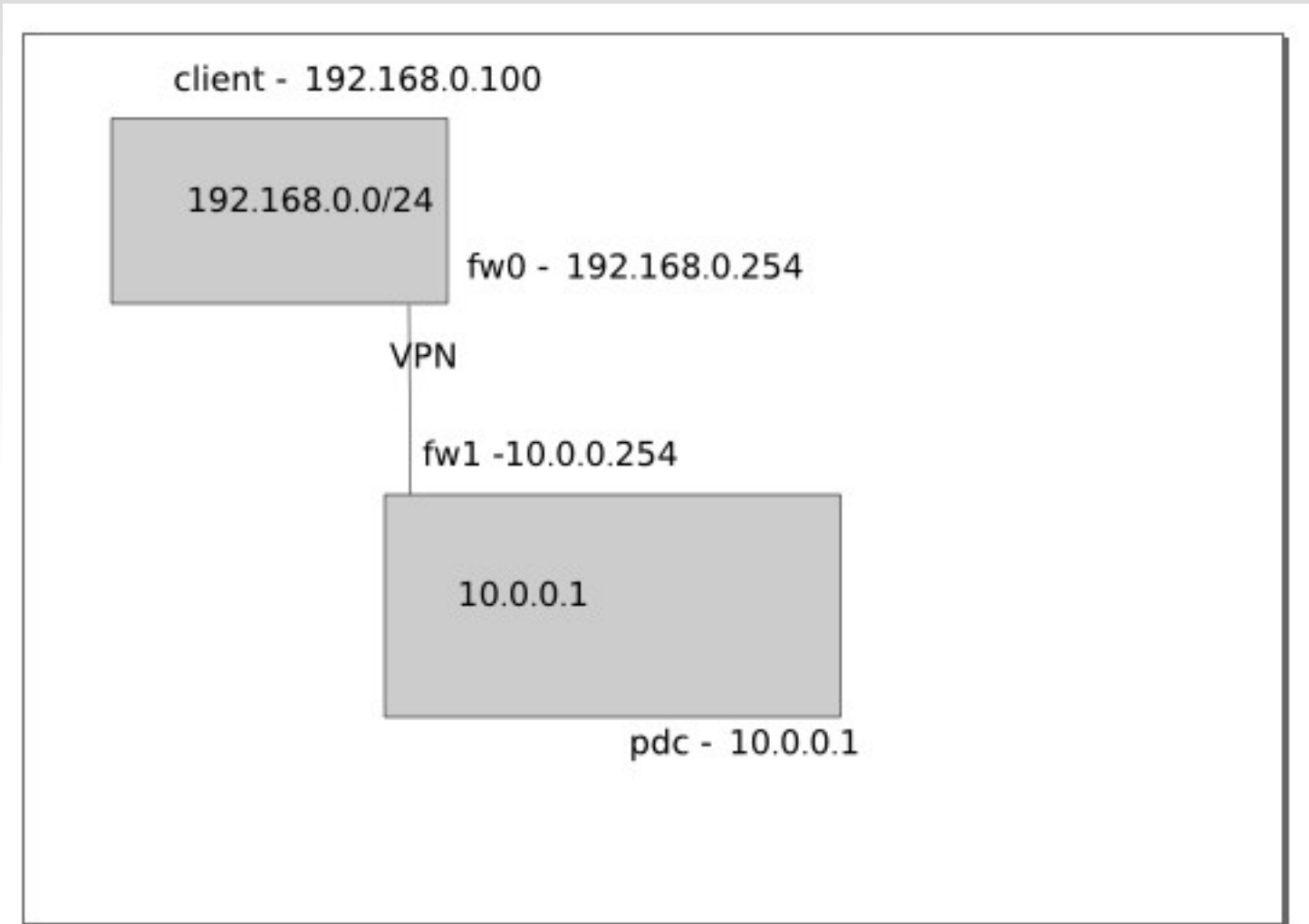
- Netkit è un sistema basato su UML che facilita il setup di un singolo pc ed in modo più eclatante di una rete
- fornisce alcuni comandi (script shell) che creano singole macchine, permettono di configurarle o fanno partire intere sottoreti, permettendo di salvare i risultati da una volta all'altra.
- E` sviluppato all'università roma3

Dominio di collisione

- ogni interfaccia può partecipare ad un dominio di collisione a scelta:
 - quello della macchina host
 - uno (o più) differenti
- è quindi possibile fare simulazioni di sottoreti differenti



Esempio



LTSP – la basi dhcp

- Abbiamo usato alcuni LTSP client e sappiamo che richiedono alcune informazioni al server dhcp: vediamo nel dettaglio la fase di boot e le implicazioni nella configurazione del dhcp-server

Cosa passa

- DNS
- IP / netmask
- Wins server
- Kernel: il kernel viene poi passato da un servizio che si chiama tftp che viene configurato in /etc/inetd.conf (un superserver che ascolta e passa a chi se ne deve occupare)
- E` possibile intervenire e cambiare il destino della macchina (thin/fat) sulla base di scelte fatte dal dhcp

Quali scelte?

- Che criteri abbiamo?
 - MAC address (es.: 00:50:FC:09:49:8A)
 - Marca della scheda
 - Parametri passati dalla scheda
- A noi tocca compilare l' elenco di questi MAC ed il relativo kernel che vogliamo accoppiato.

2 step

- La configurazione è fatta in due step:
 - MAC → tag
 - Tag → kernel
- Esempio di tag:
 - fat_srv1
 - thin_srv2

1[^] step: dhcp-host

- Il linguaggio del server è così (qui viene anche definito ip):
`dhcp-host=00:0c:29:10:a6:45,192.168.96.199,vm-client-ltsp1,net:ns1`
- `net:tag` è la sintassi per il tag
- Molti elementi possono esserci o meno (es.: IP, es.: nome client)
- Per completezza ho messo quello che ha tutto

2[^] step: dhcp-boot

- Il secondo step prevede l' abbinamento di un tag con un kernel:

dhcp-boot=net:thin_srv1,/ltsp/i386/pxelinux.0,ltsp-server,192.168.96.222

TFTP

- A questo punto il dhcp passa la palla
 - al kernel, che a sua volta indica un successore
 - initrd, che a sua volta passa la palla alla root via rete
 - Nbd. Questa era tradizionalmente fatta con NFS ora è stata sostituita da questo servizio che gira su una porta indicata in /etc/inetd.conf

/var/lib/tftpboot/ltsp

- Il server tftp serve file in /var/lib/tftpboot (configurato in /etc/inetd.conf):

```
i386/
|-- System.map-2.6.24-16-generic
|-- abi-2.6.24-16-generic
|-- config-2.6.24-16-generic
|-- initrd.img -> initrd.img-2.6.24-16-generic
|-- initrd.img-2.6.24-16-generic
|-- initrd.img-2.6.24-16-generic.bak
|-- nbi.img -> nbi.img-2.6.24-16-generic
|-- nbi.img-2.6.24-16-generic
|-- pxelinux.0
|-- pxelinux.cfg
| |-- default
| `-- default~
|-- vmlinuz -> vmlinuz-2.6.24-16-generic
`-- vmlinuz-2.6.24-16-generic
```

pxelinux

```
root@galileo:/var/lib/tftpboot/ltsp# cat i386/pxelinux.cfg/default  
DEFAULT vmlinuz ro initrd=initrd.img quiet splash nbdport=2000
```

```
2000 stream tcp nowait nobody /usr/sbin/tcpd /usr/sbin/  
nbdrootd /opt/ltsp/images/i386.img
```

Non una directory ma un singolo file!!!

NFS vs NBD

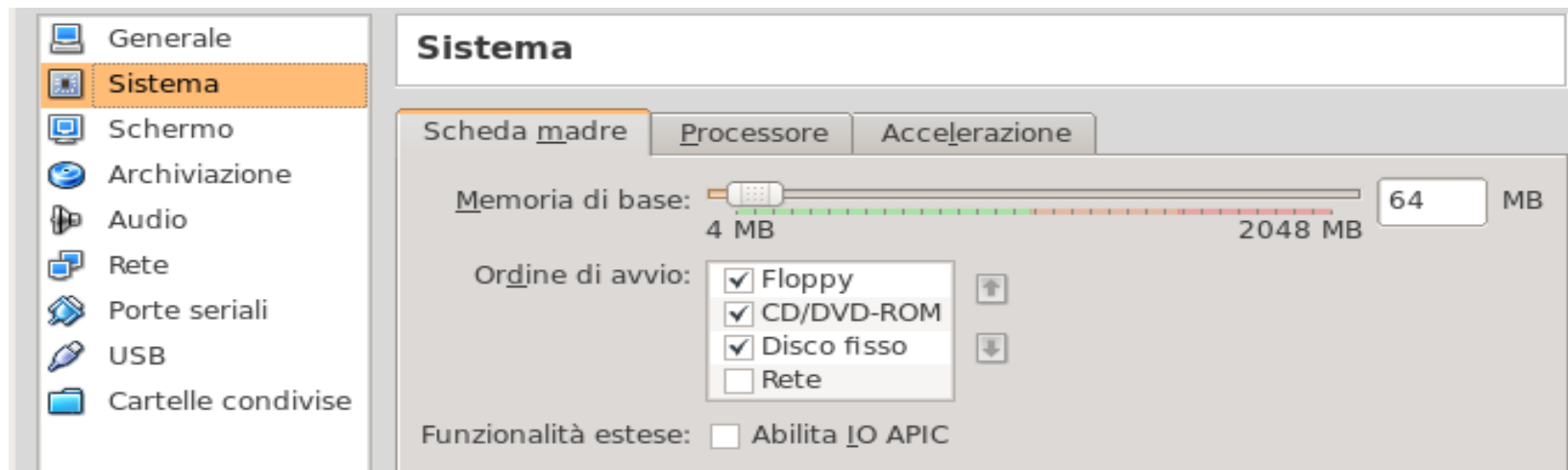
- Per analogia, la /home viene servita dal server con un altro meccanismo, ovvero via NFS.
- Debian usa ancora NFS perché è più semplice (non serve fare Itsp-update-image) anche se pare meno veloce

Come capire cosa passa il server dhcp?

- Creiamo una macchina VirtualBox con solo un gPXE che risponde
- Il gPXE (analogamente ad etherboot) è un software che emula PXE Preboot eXecution Environment), ovvero fa esattamente quello (e più di quello) che farebbe la scheda hardware, ma... senza le sue limitazioni

Macchina gPXE

- Facciamo una macchina senza dischi
- Mettiamo un floppy con immagine il file gpxe...
- Alla partenza della macchina virtuale (che impostiamo per avere un boot da floppy):
control-b + dhcp net0 + config



config

```
gPXE option configuration console

keep-san..... <not specified>_
hostname..... <not specified>
filename..... /ltsp/ifat386/pxelinux.0
root-path..... <not specified>
username..... <not specified>
password..... <not specified>
priority..... <not specified>
ip..... 192.168.50.130
netmask..... 255.255.255.0
gateway..... 192.168.50.1
mac..... 08:00:27:9a:ab:2b
busid..... 01:10:22:20:00
initiator-iqn.. <not specified>
reverse-username <not specified>
reverse-password <not specified>
dhcp-server.... 192.168.50.1
user-class..... <not specified>
keep-san - Preserve SAN connection
dns..... 192.168.96.254
Ctrl-X - exit configuration utility
next-server.... 192.168.96.222
```

Problema vero

- Analizziamo un problema vero: il client che non funziona al piano di sotto!



Cambiamo MAC

- Da VirtualBox è possibile cambiare il MAC ed arrivare quindi a macchine formalmente differenti per il server
- Cambiamo la configurazione del server (via interfaccia web) e vediamo se passa correttamente i valori al client