



Università degli Studi Roma Tre  
Dipartimento di Informatica e Automazione  
Computer Networks Research Group

# netkit lab

## single-host

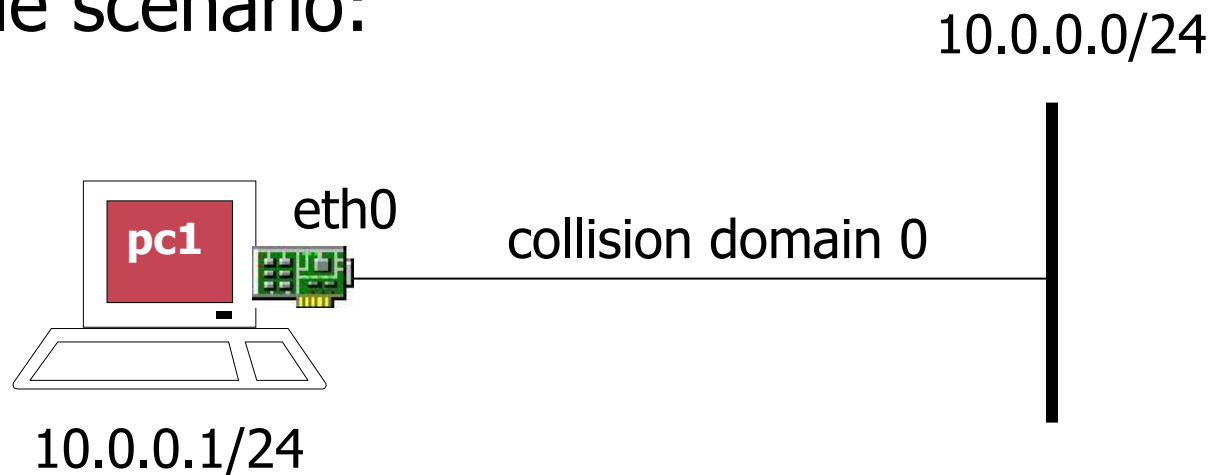
<b>Version</b>	2.2
<b>Author(s)</b>	G. Di Battista, M. Patrignani, M. Pizzonia, M. Rimondini
<b>E-mail</b>	contact@netkit.org
<b>Web</b>	<a href="http://www.netkit.org/">http://www.netkit.org/</a>
<b>Description</b>	how to set up and manage a single virtual machine

# copyright notice

- All the pages/slides in this presentation, including but not limited to, images, photos, animations, videos, sounds, music, and text (hereby referred to as “material”) are protected by copyright.
- This material, with the exception of some multimedia elements licensed by other organizations, is property of the authors and/or organizations appearing in the first slide.
- This material, or its parts, can be reproduced and used for didactical purposes within universities and schools, provided that this happens for non-profit purposes.
- Information contained in this material cannot be used within network design projects or other products of any kind.
- Any other use is prohibited, unless explicitly authorized by the authors on the basis of an explicit agreement.
- The authors assume no responsibility about this material and provide this material “as is”, with no implicit or explicit warranty about the correctness and completeness of its contents, which may be subject to changes.
- This copyright notice must always be redistributed together with the material, or its portions.

# single host

- netkit *little by little*: just a single virtual machine
- suggestion: before setting up a netkit lab, always make a clear diagram of the scenario that you are going to emulate!
- a simple scenario:



# step 1 – creating a virtual machine

host machine

list currently active vms

```
user@localhost:~$ vlist
```

USER	VHOST	PID	SIZE	INTERFACES
Total virtual machines:		0 (you),	0	(all users).
Total consumed memory:		0 KB (you),	0 KB	(all users).

```
user@localhost:~$ vstart pc1 --eth0=A
```

start a vm ...

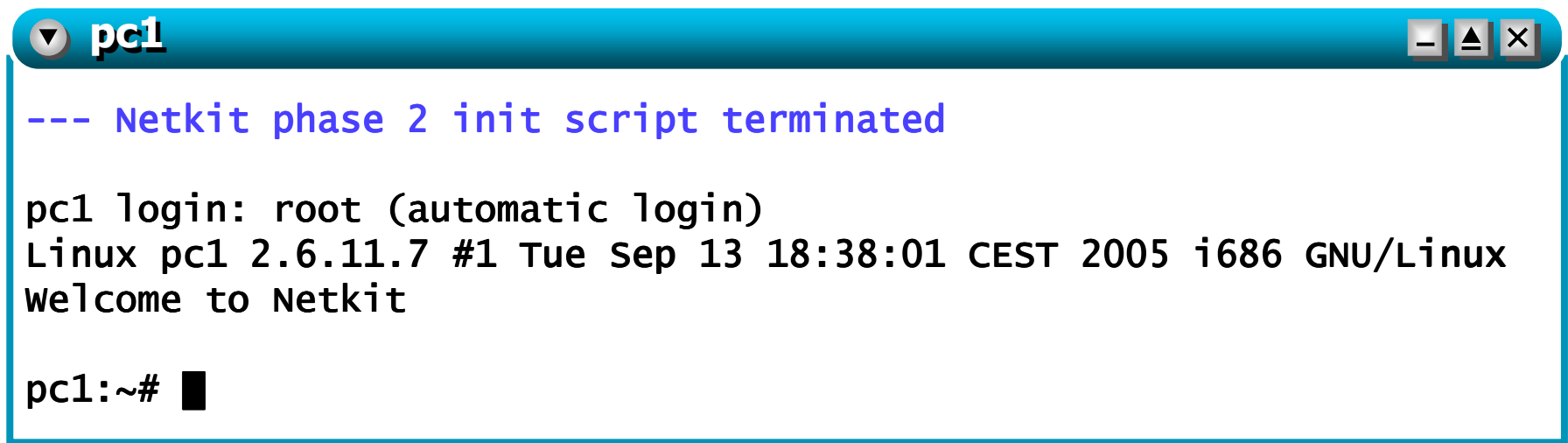
...whose name is `pc1`...

...that has a network interface on the collision domain called "A"

a virtual filesystem for `pc1` is automatically created; it will be stored inside `pc1.disk` on the host machine

## step 2 – logging on pc1

- a window containing `pc1`'s console pops up
- once the bootstrap of `pc1` is terminated, a shell prompt is automatically displayed



```
pc1
--- Netkit phase 2 init script terminated

pc1 login: root (automatic login)
Linux pc1 2.6.11.7 #1 Tue Sep 13 18:38:01 CEST 2005 i686 GNU/Linux
Welcome to Netkit

pc1:~# █
```

- now you are the administrator (root) of pc1

# step 3 – back to the host machine console

The screenshot shows a terminal window titled "host machine" with standard window controls. The first command executed is `vlist`, which displays the following table:

USER	VHOST	PID	SIZE	INTERFACES
user	pc1	2550	12380	eth0 @ A

Below the table, the following summary information is shown:

```
Total virtual machines:      1 (you),      1 (all users).
Total consumed memory: 12380 KB (you), 12380 KB (all users).
```

The second command executed is `ls -l *.disk`, which displays the following file listing:

```
-rw-r--r--  1 user group 630358016 2006-02-02 16:07 pc1.disk
```

Yellow callout boxes provide annotations for the terminal output:

- "list currently active vms" points to the `vlist` command.
- "list vm filesystems" points to the `ls -l *.disk` command.
- "file name" points to the filename `pc1.disk` in the listing.
- "user" points to the `user` field in the listing.
- "size (actual disk usage is smaller)" points to the `630358016` field.
- "update time" points to the `2006-02-02 16:07` field.

# step 4 – configuring the network interface of pc1

```
pc1:~# ifconfig eth0 10.0.0.1 netmask 255.255.255.0 broadcast 10.0.0.255 up

pc1:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FE:FD:0A:00:00:01
          inet addr:10.0.0.1  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:aff:fe00:1/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          bytes:0 (0.0 b)  TX bytes:238 (238.0 b)
          Interrupt:5

pc1:~# █
```

automatically assigned mac address

interface status

# step 5 – checking the routing table

- the routing table has been automatically updated when the interface has been brought up:

```
pc1
pc1:~# route
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
10.0.0.0         *                255.255.255.0  U         0      0      0 eth0
pc1:~# █
```

next hop

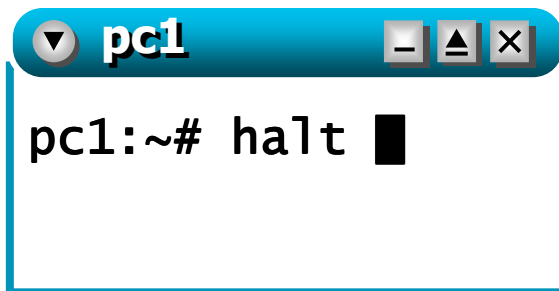
the if status is **UP**

- other labs show how to manually alter the routing table



# step 6 – shutting down the vm

- three possibilities
  - from inside



A terminal window titled 'pc1' with standard window controls (minimize, maximize, close). The terminal content shows the command 'halt' being entered at the prompt 'pc1:~#'. A black cursor is visible at the end of the command.

```
pc1:~# halt █
```

- from outside



A terminal window titled 'host machine' with standard window controls. The terminal content shows the command 'vhalt pc1' being executed at the prompt 'user@localhost:~\$'. The output is 'Halting virtual machine "pc1" (PID 3559) owned by user [..... ]'. A black cursor is visible at the end of the second prompt.

```
user@localhost:~$ vhalt pc1
Halting virtual machine "pc1" (PID 3559) owned by user [..... ]
user@localhost:~$ █
```

# step 6 – shutting down the vm

- from outside, brute force

```
host machine
user@localhost:~$ vcrash pc1

===== Crashing virtual machine "pc1" (PID 4830) =====
Virtual machine owner: user
Virtual machine mconsole socket: /home/user/.netkit/mconsole/pc1/mconsole
Crashing... done.
user@localhost:~$ █
```

- unless you chose to use vcrash, `pc1`'s filesystem is still stored in file `pc1.disk`, so it will be used again when `pc1` is restarted

# step 7 – a permanent configuration

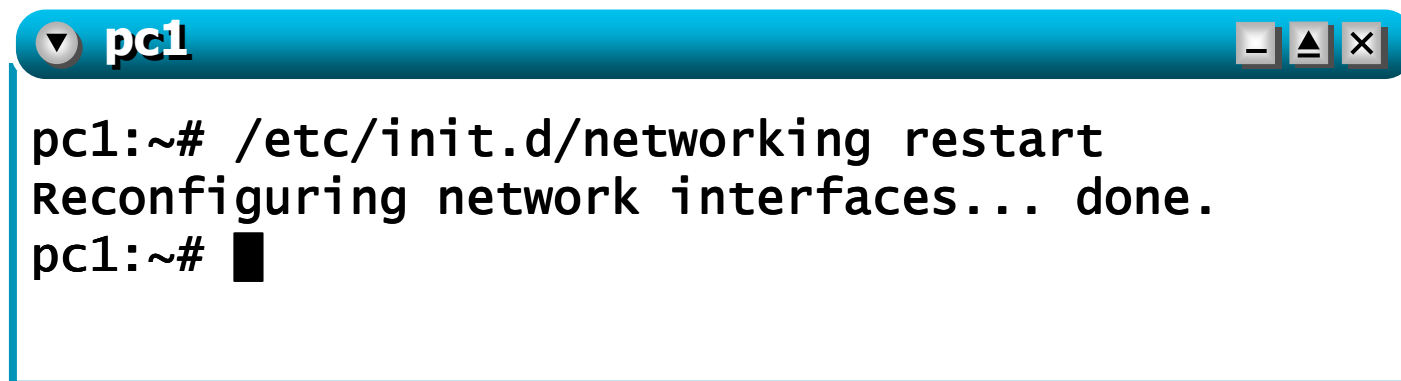
- after halting `pc1`, if you want to restart it you also have to reconfigure its network interface `eth0`
- a permanent configuration can be obtained, e.g., by editing `/etc/network/interfaces` (inside the vm `pc1`) and appending the following lines:

```
auto eth0
iface eth0 inet static
    address 10.0.0.1
    network 10.0.0.0
    netmask 255.255.255.0
```

- tips:
  - you can use an editor like `vi` or `mcedit`
  - the permanent settings of a vm can be configured inside the same files that would be used in a real linux box
- removing the vm filesystem (`pc1.disk`) removes any permanent configuration as well

# step 8 – restarting network services

- at next boot `pc1` will be automatically configured by the os which will perform the suitable `ifconfig` and `route` commands based on the contents of `/etc/network/interfaces`
- the new configuration can also be fetched without rebooting by restarting network services:

A terminal window titled 'pc1' with standard window controls (minimize, maximize, close). The terminal shows the command `/etc/init.d/networking restart` being executed, followed by the output `Reconfiguring network interfaces... done.` and a new prompt `pc1:~#` with a cursor.

```
pc1:~# /etc/init.d/networking restart
Reconfiguring network interfaces... done.
pc1:~# █
```