Fondamenti di Internet: “Aspetti pratici delle reti di telecomunicazioni”
First Part
Outline

• First part:
  ✓ Linux for dummies
  ✓ Emulation of computer networks: Netkit

• Second part:
  ✓ Configuration of network interfaces: IProute2
  ✓ Analyzing of network traffic: Wireshark & Tcpdump
  ✓ Examples of network traffic: Scapy, Ping
Why Linux?

• Open Source Philosophy: no secret for the user
• Widely used in networks
  ✓ Routers
  ✓ Embedded systems
  ✓ Servers
• Free distribution
• Large support and documentation
• ...and “User Mode Linux”
Linux in a nutshell

Kernel
- Driver
- Scheduler
- FS
- Net
- Shell

Operating System
- glibc
- GCC
- Base Utils (GNU)

Programs
- Drivers
- Games
- Office (e.g. OpenOffice)
- Graphics (e.g. Gimp)
- Others...
- Packet Manager (apt, portage, rpm...)

Distributions
- Slackware
- Debian
- Ubuntu
- Mandriva
- Arch
- Fedora
- Suse
- Red Hat
- Gentoo
- Live Distro (e.g. Knoppix)

Desktop Environment (KDE, Gnome...)
Services/daemons
Packet Manager (apt, portage, rpm...)

Linux (typical) directory tree
## Usefull shell commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ls</td>
<td>“List” file/directory contained in the current directory. Usefull options: “–al”</td>
</tr>
<tr>
<td>mkdir &lt;NEW_DIRECTORY&gt;</td>
<td>Create a new directory</td>
</tr>
<tr>
<td>rm &lt;FILE&gt;</td>
<td>Remove file. Usefull options “-rf” to remove also directory</td>
</tr>
<tr>
<td>touch &lt;FILE&gt;</td>
<td>Create a new, empty, file</td>
</tr>
<tr>
<td>nano &lt;FILE&gt;</td>
<td>edit a file (or create if it does not exist)</td>
</tr>
<tr>
<td>cp &lt;ORIG&gt; &lt;DEST&gt;</td>
<td>copy a file. Usefull options “-r” to copy also directory</td>
</tr>
<tr>
<td>mv &lt;ORIG&gt; &lt;DEST&gt;</td>
<td>move or rename a file/directory</td>
</tr>
</tbody>
</table>
Support

The Magic Word man

Documentation
Linux User Guide
Distro Specific guides (e.g. http://help.ubuntu-it.org/)

Linux User Group
http://www.torlug.org

Download the netkit live distribution:
Linux and Networking

User Space
System calls: Socket

Kernel Space

Drivers
Hardware

Application
Transport
Network
Data Link
Physical

Interface Mapping

eth0
eth1
Outline

- First part:
  - ✓ Linux for dummies
  - ✓ Emulation of computer networks: Netkit

- Second part:
  - ✓ Configuration of network interfaces: IProute2
  - ✓ Analyzing of network traffic: Wireshark & Tcpdump
  - ✓ Examples of network traffic: Scapy, Ping
Simulation vs. Emulation

Emulation and simulation systems put at user’s disposal a virtual environment that can be exploited for tests, experiments, measures

**Simulation**

Simulation systems aim at reproducing the performance of a real-life system (latency time, packet loss, etc.)

- e.g.: ns, omnet++, ...

**Emulation**

Emulation systems aim at accurately reproducing the functionalities of a real-life system (configurations, architectures, protocols), with limited attention to performance

Netkit

"The poor man's system to experiment computer networking"

**WHAT'S NETKIT:**

- a system for emulating computer networks
- based on uml (user-mode linux)
  - user-mode linux is a linux kernel (inner part of the linux os) that can be executed as a user process on a standard linux box
  - a user-mode linux process is also called virtual machine (vm), while the linux box that hosts a virtual machine is called host machine (host)
- each emulated network device is a virtual linux box
- note: the linux os is shipped with software supporting most of the network protocols
  - hence, any linux machine can be configured to act as a bridge/switch or as a router
Netkit Virtual Machine Commands

vstart: starts a new virtual machine
vlist: lists currently running virtual machines
vconfig: attaches network interfaces to running vms
vhalt: gracefully halts a virtual machine
vcrash: causes a virtual machine to crash
vclean: “panic command” to clean up all netkit processes (including vms) and configuration settings on the host machine

More Info about netkit commands? man vstart or visit wiki.netkit.org
Setting-up a Netkit Lab

Netkit Lab: automatize multiple virtual machine startup. To create a lab we need:

- a lab configuration file describing the network topology (lab.conf)
- a set of subdirectories that contain the configuration settings for each virtual machine
- [optionally] .startup and .shutdown files that describe actions performed by virtual machines when they are started or halted
- [optionally] a lab.depfile describing dependency relationships on the startup order of virtual machines
Example of lab file: mylab.conf

pc1[0]=A
pc2[0]=A
pc2[1]=B
pc2[mem]=256
pc3[0]=B
Netkit Lab Commands

lstart: starts a netkit lab
lhalt: gracefully halts all vms of a lab
lcrash: causes all the vms of a lab to crash
lclean: removes temporary files from a lab directory
linfo: provides information about a lab without starting it
ltest: allows to run tests to check that the lab is working properly

As for the case of virtual machine, for help type man lstart or visit wiki.netkit.org
File Exchange and Internet Connection

the directory `/hosthome` inside a virtual machine directly points to the home directory of the current user on the real host

The directory `/hostlab` is shared inside a lab

vstart can automatically configure tunnels ("tap interfaces") by which a virtual machine can access an external network
File .startup

1. For each VM is possible to create a “VM_NAME”.startup file
2. VM_NAME is the name of the virtual machine
   Example: pc1 → pc1.startup
3. The commands written in the .startup file will be executed at the end of the boot process
TAP interface

1. A VM interface can be attached to a “tap” instead of a collision domain
2. In this case, a virtual (tap) interface will be created on the host machine and will be connected to the specific VM interface

Example (in lab.conf)
pc1[0]=tap,10.0.0.1,10.0.0.2

In this case the eth0 interface on pc1 will be connected to a tap on the host machine with address 10.0.0.1 while eth0 on pc1 will be configured automatically with address 10.0.0.2
Second Part
Outline

• First part:
  ✓ Linux for dummies
  ✓ Emulation of computer networks: Netkit

• Second part:
  ✓ Configuration of network interfaces: IProute2
  ✓ Analyzing of network traffic: Wireshark & Tcpdump
  ✓ Examples of network traffic: Scapy, Ping
IPRoute2

Iproute2 is a collection of utilities for controlling TCP / IP networking and traffic control in Linux. [1]

Implemented on almost any linux system with kernel > 2.2.X

Mangle almost any network stuff:
- Interface setting up and drop down
- Routing Tables
- Arp cache
- And more...
  - Multiple routing tables
  - Policy Routing
  - Tunnels
  - NAT
  - IPsec policy and associations

Replace “old” (but still used) commands: route, arp, ifconfig


Simple and basic guide: http://www.commedia.it/ccontavalli/docs-it/ip/ip4dummies.pdf
IPRoute2: data link settings of a network interface

IPRoute2 command to manage data link settings

how to? # ip link help

```
ip link help
Usage: ip link set DEVICE { up | down } |
       arp { on | off } |
       dynamic { on | off } |
       multicast { on | off } |
       allmulticast { on | off } |
       promisc { on | off } |
       trailers { on | off } |
       txqueuelen PACKETS |
       name NEWNAME |
       address LLADDR | broadcast LLADDR |
       mtu MTU }
ip link show [ DEVICE ]
```
IProute2: ip link <...>, a basic usage

In a UNIX-like system, a network interface can have different names. Under Linux-based operating systems: eth0, eth1, eth2, etc. Generally ethX

Activating a network interface

# ip link set eth0 up

De-activating a network interface

# ip link set eth0 down

Changing the data link address (mac address)

# ip link set eth0 address xx:xx:xx:xx:xx:xx

Showing data link attributes of system network interfaces

# ip link show
IProute2: network (IP) settings of a network interface

IProute2 command to manage network (IP, Internet Protocol) settings

how to? # ip address help

```
ip address help
Usage: ip addr {add|del} IFADDR dev STRING
       ip addr {show|flush} [ dev STRING ] [ scope SCOPE-ID ]
                        [ to PREFIX ] [ FLAG-LIST ] [ label PATTERN ]
IFADDR := PREFIX | ADDR peer PREFI
          [ broadcast ADDR ] [ anycast ADDR ]
           [ label STRING ] [ scope SCOPE-ID ]
SCOPE-ID := [ host | link | global | NUMBER ]
FLAG-LIST := [ FLAG-LIST ] FLAG
FLAG := [ permanent | dynamic | secondary | primary | tentative | deprecated ]
```

We will manage it better in the next lectures
Outline

• First part:
  ✓ Linux for dummies
  ✓ Emulation of computer networks: Netkit

• Second part:
  ✓ Configuration of network interfaces: IProute2
  ✓ Analyzing of network traffic: Wireshark & Tcpdump
  ✓ Examples of network traffic: Scapy, Ping
Wireshark: “the” network analyzer

Wireshark is the world's foremost network protocol analyzer, and is the de facto (and often de jure) standard across many industries and educational institutions.

Wireshark development thrives thanks to the contributions of networking experts across the globe. It is the continuation of a project that started in 1998 (Ethereal).
* Deep inspection of hundreds of protocols, with more being added all the time
* Live capture and offline analysis
* Standard three-pane packet browser
* Multi-platform: Runs on Windows, Linux, OS X, Solaris, FreeBSD, NetBSD, and many others
* Captured network data can be browsed via a GUI, or via the TTY-mode TShark utility
* The most powerful display filters in the industry
* Rich VoIP analysis
* Read/write many different capture file formats
* Capture files compressed with gzip can be decompressed on the fly
* Live data can be read from Ethernet, IEEE 802.11, PPP/HDLC, ATM, Bluetooth, USB, Token Ring, Frame Relay, FDDI, and others (depending on your platform)
* Decryption support for many protocols, including IPsec, ISAKMP, Kerberos, SNMPv3, SSL/TLS, WEP, and WPA/WPA2 (having the keys 😊)
* Coloring rules can be applied to the packet list for quick, intuitive analysis
* Output can be exported to XML, PostScript®, CSV, or plain text
Tcpdump: command line network analyzer

```
tcpdump [ -adeflnNOpqRStuvxX ] [ -c count ]
  [ -C file_size ] [ -F file ]
  [ -i interface ] [ -m module ] [ -r file ]
  [ -s snaplen ] [ -T type ] [ -w file ]
  [ -E algo:secret ] [ expression ]
```

Why using tcpdump? Netkit hasn’t a graphical environment, using tcpdump to capture and Wireshark to display pkts.
Outline

• First part:
  ✓ Linux for dummies
  ✓ Emulation of computer networks: Netkit

• Second part:
  ✓ Configuration of network interfaces: IProute2
  ✓ Analyzing of network traffic: Wireshark & Tcpdump
  ✓ Examples of network traffic: Scapy, Ping
- What is SCAPY? Scapy is a powerful interactive packet manipulation program. It is able to forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more.

http://www.secdev.org/projects/scapy/

- What is PING? Ping is a computer network tool used to test whether a particular host is reachable across an IP network; it is also used to self test the network interface card of the computer, or as a speed test. It works by sending ICMP “echo request” packets to the target host and listening for ICMP “echo response” replies.

man ping 😊
Example: Lab2

Collision domain A

- pc1
  - eth0: 10.0.0.1/16

- pc2
  - eth0: 10.0.0.2/16

- Host machine
  - tap: 192.168.0.1/16
  - eth1: 192.168.0.2/16