



REDES

*Grados Ing. Informática / Ing. de Computadores / Ing. del Software / Doble Grado
Universidad Complutense de Madrid*

Introducción al Laboratorio de Redes

Profesoras:

Sandra Catalán Pallarés

Guadalupe Miñana Roperó



REDES

*Grados Ing. Informática / Ing. de Computadores / Ing. del Software / Doble Grado
Universidad Complutense de Madrid*

Descripción del Framework

Laboratorio Virtual

- **Objetivo**

- Proporcionar un entorno virtual portable para el desarrollo de las prácticas

- **Elementos**

- **Máquina Host (en el laboratorio o en casa)**

- Ordenador físico que funciona como máquina host
 - SO (en el laboratorio) → Ubuntu GNU/Linux
 - SO (en casa) → Linux, Windows, o Mac OSX
 - VirtualBox instalado

- **Máquina virtual principal**

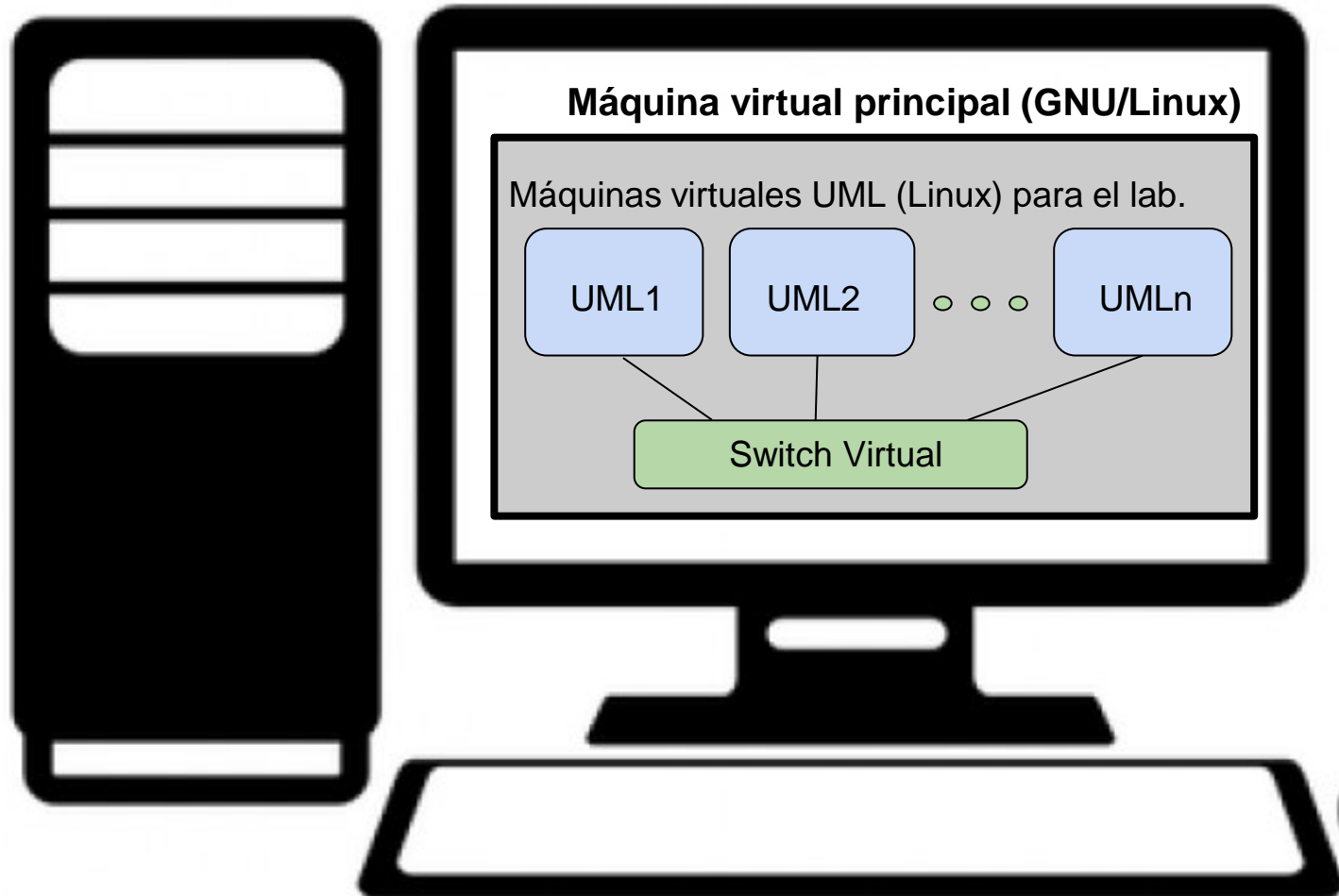
- Máquina virtual que contiene el entorno del laboratorio
 - Se ejecuta como guest en la máquina física utilizando VirtualBox
 - Basada en Debian GNU/Linux OS con interfaz gráfica
- Incluye las herramientas necesarias para desarrollar todas las prácticas
 - User Mode Linux (UML): virtualización que contiene las máquinas virtuales del laboratorio
 - Switches virtuales para interconectar las máquinas virtuales del laboratorio

- **Máquinas virtuales UML del laboratorio**

- Máquinas Linux simples para realizar las prácticas de laboratorio
 - Se ejecutan como guests en la máquina virtual principal usando UML
 - Sólo tienen línea de comandos (sin interfaz gráfica)
 - Funcionan como hosts de red o routers para las prácticas de laboratorio

Laboratorio Virtual

Máquina Física Host (con VirtualBox)



Laboratorio Virtual

- **Arrancar la máquina virtual principal en el laboratorio**

1. Arrancar el ordenador del laboratorio usando el SO GNU/Linux y haciendo login con el usuario “usuario VMs”
2. Abrir una terminal de linux y ejecutar los siguientes scripts para limpiar el entorno y arrancar la máquina virtual principal:

```
$ redeslab del  
$ redeslab regenerate  
$ redeslab start
```
1. Hacer log in en la máquina virtual principal usando el nombre de usuario redes y la contraseña cursoredes.

- **Arrancar la máquina virtual principal en casa**

1. Instalar VirtualBox y el Extension Pack de VirtualBox
 - Ir a la página <https://www.virtualbox.org/wiki/Downloads>
 - Descargar e instalar el paquete de VirtualBox correspondiente a tu SO
 - Descargar e instalar el Oracle VM VirtualBox Extension Pack
1. Descargar la imagen de la máquina virtual principal de este [link](#), llamada AR_RNG_2021.ova
2. Hacer clic dos veces en el archivo AR_RNG_2021.ova para instalar la máquina virtual principal en VirtualBox
3. Abrir la consola de VirtualBox y arrancar la máquina virtual principal AR_RNG_2021
4. Hacer log in en la máquina virtual principal con el usuario redes y la contraseña cursoredes.

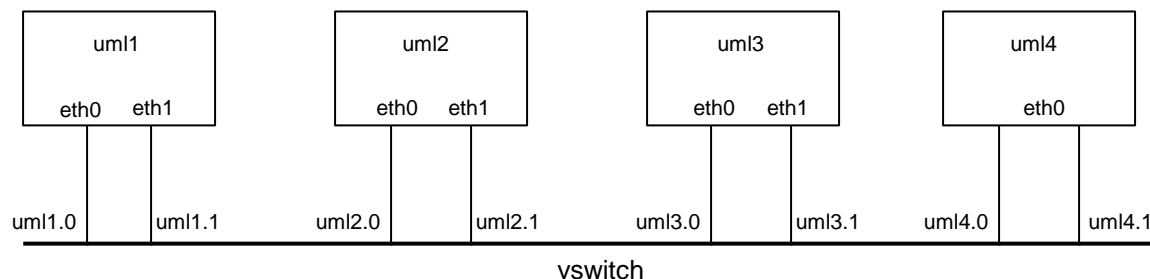
Laboratorio Virtual

- **Desplegar las máquinas virtuales UML**

(Esto se tiene que hacer en la máquina virtual principal)

1. Crear un fichero “net.conf” con la topología de red

- Ejemplo:



```
$ nano net.conf
```

```
defsw vswitch uml1.0 uml1.1 uml2.0 uml2.1 uml3.0 uml3.1 uml4.0 uml4.1
```

- Truco: En las sesiones de laboratorio, podemos usar siempre la misma topología, así que el proceso se puede simplificar creando el fichero net.conf al principio de cada sesión con el siguiente comando:

```
$ echo defsw vswitch uml{1..5}.{0..3} > net.conf
```

2. Desplegar la topología

```
$ sudo ifovsdel
```

```
$ sudo ifovsparse net.conf
```

3. Arrancar las máquinas virtuales UML (usando el script “lanza”)

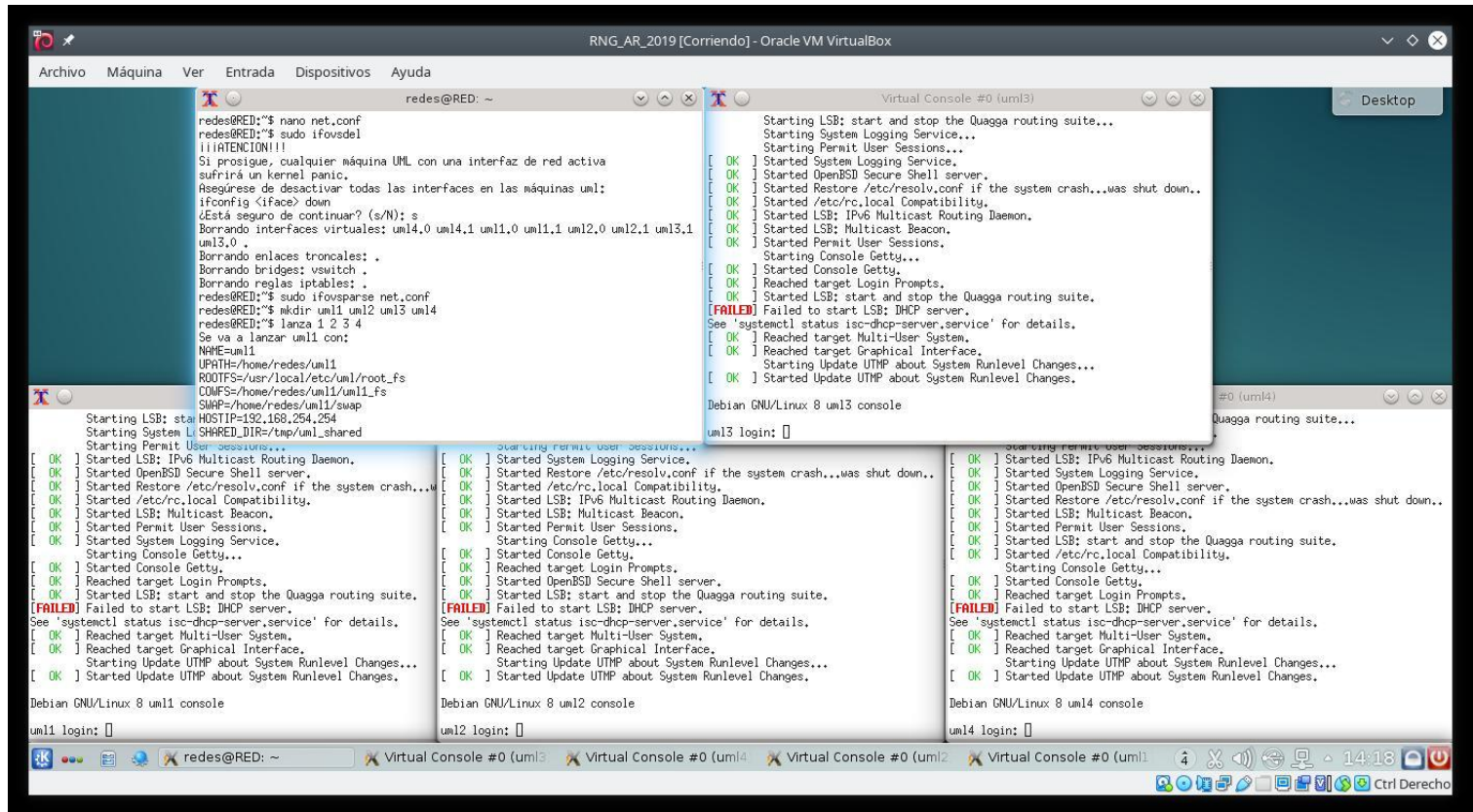
```
$ mkdir uml1 uml2 uml3 uml4
```

```
$ lanza 1 2 3 4
```

Laboratorio Virtual

- **Trabajando con máquinas virtuales UML**

- Hacer login in en las máquinas virtuales UML usando el usuario root (sin contraseña)
- Las máquinas virtuales UML solo tienen línea de comandos (sin interfaz gráfica)



```
redes@RED: ~  
redes@RED:~$ nano net.conf  
redes@RED:~$ sudo ifovsdel  
[!] ATENCION!!!  
Si prosigue, cualquier máquina UML con una interfaz de red activa  
sufrirá un kernel panic.  
Asegúrese de desactivar todas las interfaces en las máquinas uml:  
ifconfig <iface> down  
¿Está seguro de continuar? (s/N): s  
Borrando interfaces virtuales: uml4.0 uml4.1 uml1.0 uml1.1 uml2.0 uml2.1 uml3.1  
uml3.0 .  
Borrando enlaces troncales: .  
Borrando bridges: vswitch .  
Borrando reglas iptables: .  
redes@RED:~$ sudo ifovsparse net.conf  
redes@RED:~$ mkdir uml1 uml2 uml3 uml4  
redes@RED:~$ lanza 1 2 3 4  
Se va a lanzar uml1 con:  
NAME=uml1  
UPATH=/home/redes/uml1  
ROOTFS=/usr/local/etc/uml/root.fs  
CDWFS=/home/redes/uml1/uml1_fs  
SWAP=/home/redes/uml1/swap  
HOSTIP=192.168.254.254  
Starting System L. SHARED_DIR=/tmp/uml_shared  
Starting Permit User Sessions...  
[ OK ] Started LSB: IPv6 Multicast Routing Daemon.  
[ OK ] Started System Logging Service.  
[ OK ] Started Restore /etc/resolv.conf if the system crash...was shut down..  
[ OK ] Started /etc/rc.local Compatibility.  
[ OK ] Started LSB: Multicast Beacon.  
[ OK ] Started Permit User Sessions.  
[ OK ] Started System Logging Service.  
Starting Console Getty...  
[ OK ] Started Console Getty.  
[ OK ] Reached target Login Prompts.  
[ OK ] Started LSB: start and stop the Quagga routing suite.  
[ FAILED ] Failed to start LSB: DHCP server.  
See 'systemctl status isc-dhcp-server.service' for details.  
[ OK ] Reached target Multi-User System.  
[ OK ] Reached target Graphical Interface.  
Starting Update UTMP about System Runlevel Changes...  
[ OK ] Started Update UTMP about System Runlevel Changes.  
Debian GNU/Linux 8 uml1 console  
uml1 login: [ ]  
Virtual Console #0 (uml3)  
Starting LSB: start and stop the Quagga routing suite...  
Starting System Logging Service...  
Starting Permit User Sessions...  
[ OK ] Started System Logging Service.  
[ OK ] Started OpenBSD Secure Shell server.  
[ OK ] Started Restore /etc/resolv.conf if the system crash...was shut down..  
[ OK ] Started /etc/rc.local Compatibility.  
[ OK ] Started LSB: IPv6 Multicast Routing Daemon.  
[ OK ] Started LSB: Multicast Beacon.  
[ OK ] Started Permit User Sessions.  
Starting Console Getty...  
[ OK ] Started Console Getty.  
[ OK ] Reached target Login Prompts.  
[ OK ] Started LSB: start and stop the Quagga routing suite.  
[ FAILED ] Failed to start LSB: DHCP server.  
See 'systemctl status isc-dhcp-server.service' for details.  
[ OK ] Reached target Multi-User System.  
[ OK ] Reached target Graphical Interface.  
Starting Update UTMP about System Runlevel Changes...  
[ OK ] Started Update UTMP about System Runlevel Changes.  
Debian GNU/Linux 8 uml3 console  
uml3 login: [ ]  
Virtual Console #0 (uml4)  
Starting LSB: start and stop the Quagga routing suite...  
Starting System Logging Service...  
Starting Permit User Sessions...  
[ OK ] Started System Logging Service.  
[ OK ] Started OpenBSD Secure Shell server.  
[ OK ] Started Restore /etc/resolv.conf if the system crash...was shut down..  
[ OK ] Started /etc/rc.local Compatibility.  
[ OK ] Started LSB: Multicast Beacon.  
[ OK ] Started Permit User Sessions.  
Starting Console Getty...  
[ OK ] Started Console Getty.  
[ OK ] Reached target Login Prompts.  
[ OK ] Started LSB: start and stop the Quagga routing suite.  
[ FAILED ] Failed to start LSB: DHCP server.  
See 'systemctl status isc-dhcp-server.service' for details.  
[ OK ] Reached target Multi-User System.  
[ OK ] Reached target Graphical Interface.  
Starting Update UTMP about System Runlevel Changes...  
[ OK ] Started Update UTMP about System Runlevel Changes.  
Debian GNU/Linux 8 uml4 console  
uml4 login: [ ]
```

- Cuando acabes la práctica, debes apagar las máquinas virtuales UML con el siguiente comando en cada una:

```
# shutdown -h now
```



REDES

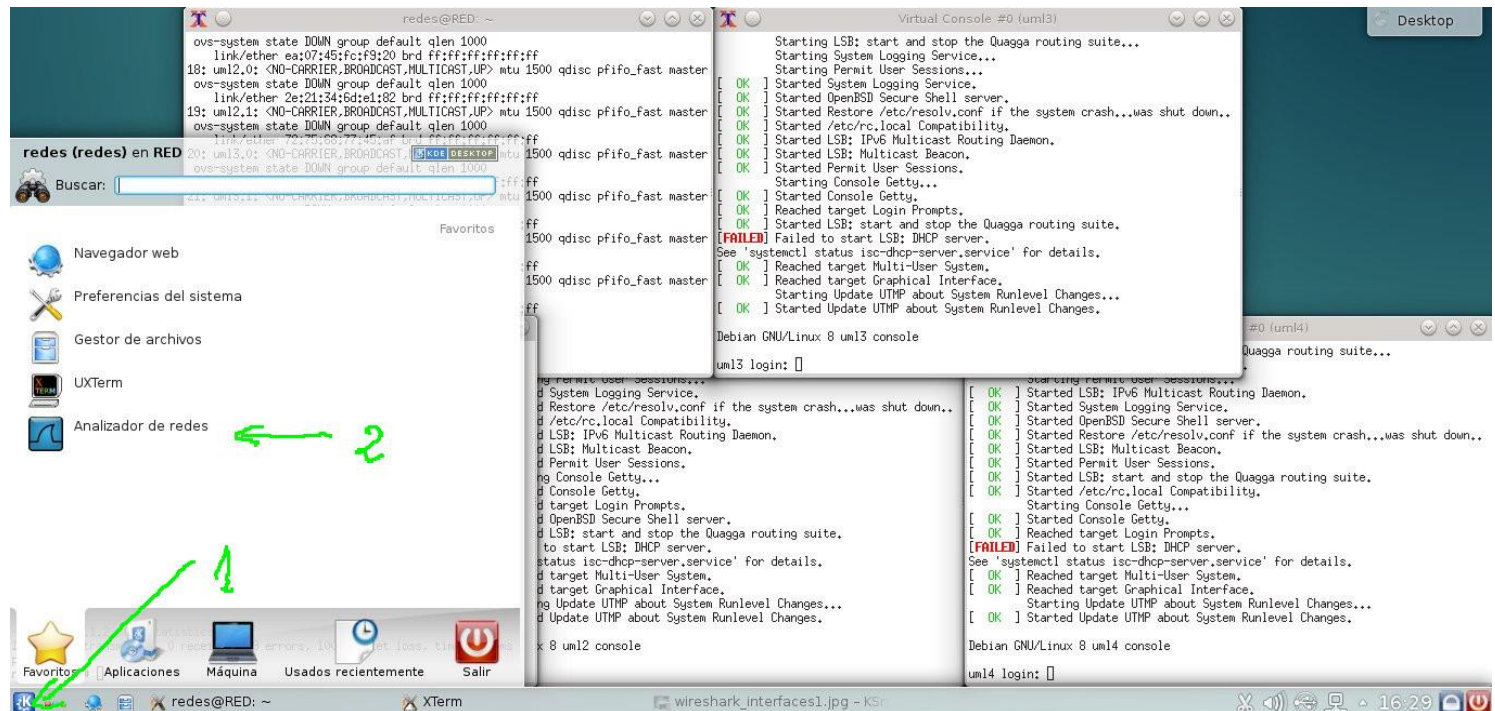
*Grados Ing. Informática / Ing. de Computadores / Ing. del Software / Doble Grado
Universidad Complutense de Madrid*

En analizador de red: wireshark

Wireshark

- **Análisis del tráfico de red con Wireshark**

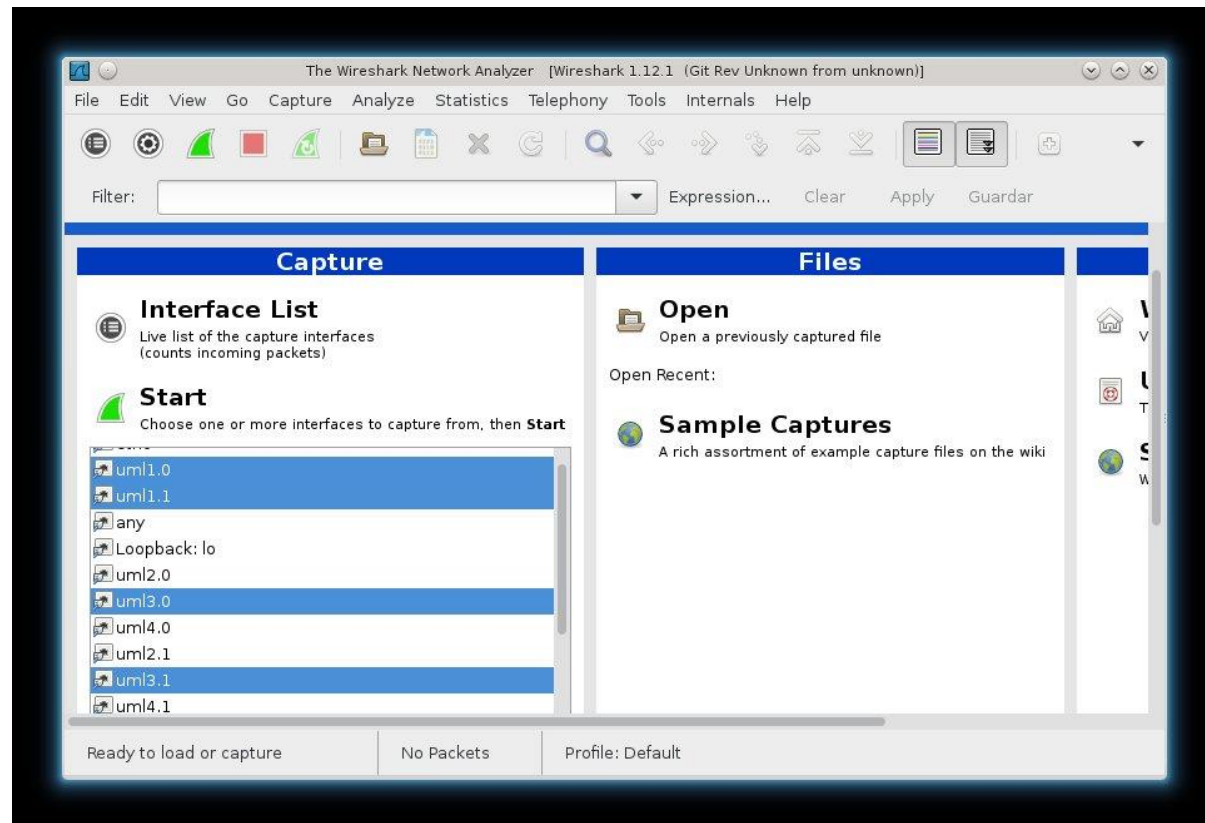
- El tráfico generado por las distintas máquinas virtuales UML puede ser capturado y analizado con Wireshark
- Wireshark es una herramienta gráfica que solo se ejecuta en la máquina virtual principal
 - Para ejecutar Wireshark haz clic en el icono correspondiente del escritorio (o puedes teclear wireshark en una consola).



Wireshark

- **Trabajando con Wireshark**

- Para mostrar el tráfico, se debe especificar la/s interfaz/ces a escuchar (por ejemplo, uml1.0 y uml1.1).
 - Para seleccionar varias interfaces, se selecciona la primera y luego las siguientes mientras se mantiene pulsada la tecla Ctrl
- Luego se hace clic en Start para empezar la captura



Wireshark

• Ejemplo de captura

The image shows the Wireshark network traffic capture interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, and Help. Below the menu is a toolbar with various icons for file operations, capture control, and analysis. A filter bar is present with a dropdown menu and buttons for Expression..., Clear, and Apply.

The main packet list displays the following data:

No. .	Time	Source	Destination	Protocol	Info
324	709.687786	192.168.0.2	192.168.0.1	ICMP	Destination unreachable (Fragmentation needed)
325	710.692665	192.168.0.1	192.168.2.4	IP	Fragmented IP protocol (proto=ICMP 0x01, off=0, ID=f1dc)
326	710.692702	192.168.0.1	192.168.2.4	ICMP	Echo (ping) request
327	710.693437	192.168.2.4	192.168.0.1	ICMP	Echo (ping) reply
328	711.694045	192.168.0.1	192.168.2.4	IP	Fragmented IP protocol (proto=ICMP 0x01, off=0, ID=f1dd)
329	711.694084	192.168.0.1	192.168.2.4	ICMP	Echo (ping) request
330	711.694836	192.168.2.4	192.168.0.1	ICMP	Echo (ping) reply
331	714.691867	02:00:00:00:02:f0	02:00:00:00:01:f0	ARP	Who has 192.168.0.1? Tell 192.168.0.2
332	714.692000	02:00:00:00:01:f0	02:00:00:00:02:f0	ARP	192.168.0.1 is at 02:00:00:00:01:f0
333	1209.652589	fe80::f86b:cff:febc:39c1	ff02::fb	MDNS	Standard query PTR_services.dns-sd._udp.local, "QM" question
334	1209.733181	fe80::984a:49ff:febc:30c	ff02::fb	MDNS	Standard query PTR_services.dns-sd._udp.local, "QM" question
335	54035.392930	fe80::f86b:cff:febc:39c1	ff02::fb	MDNS	Standard query PTR_services.dns-sd._udp.local, "QM" question
336	54035.393084	fe80::984a:49ff:febc:30c	ff02::fb	MDNS	Standard query PTR_services.dns-sd._udp.local, "QM" question
337	57635.393223	fe80::f86b:cff:febc:39c1	ff02::fb	MDNS	Standard query PTR_services.dns-sd._udp.local, "QM" question

The detailed view of the selected packet (Frame 329) shows the following information:

- Frame 329 (266 bytes on wire, 266 bytes captured)
- Ethernet II, Src: 02:00:00:00:01:f0 (02:00:00:00:01:f0), Dst: 02:00:00:00:02:f0 (02:00:00:00:02:f0)
- Internet Protocol, Src: 192.168.0.1 (192.168.0.1), Dst: 192.168.2.4 (192.168.2.4)
- Version: 4
- Header length: 20 bytes
- Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
- Total Length: 252
- Identification: 0xf1dd (61917)
- Flags: 0x00
 - 0.. = Reserved bit: Not Set
 - .0. = Don't fragment: Not Set
 - ..0 = More fragments: Not Set
- Fragment offset: 576
- Time to live: 64
- Protocol: ICMP (0x01)
- Header checksum: 0x0486 [correct]
- Source: 192.168.0.1 (192.168.0.1)
- Destination: 192.168.2.4 (192.168.2.4)
- [IP Fragments (808 bytes): #328(576), #329(232)]
- Internet Control Message Protocol
 - Type: 8 (Echo (ping) request)
 - Code: 0 ()
 - Checksum: 0xc1c1 [correct]
 - Identifier: 0x5d08
 - Sequence number: 3 (0x0003)
 - Data (800 bytes)

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0000 02 00 00 00 02 f0 02 00 00 00 01 f0 08 00 45 00 .....E..
0010 00 fc f1 dd 00 48 40 01 04 86 c0 a8 00 01 c0 a8 .....H@.....
0020 02 04 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 45 ..89;.<=>?@ABCDE
0030 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51 52 53 54 55 FGHIJKLM NOPQRSTU
0040 56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 62 63 64 65 VWXYZ[\]^_`abcde
0050 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 fghijklm nopqrstu
```

The status bar at the bottom indicates: Internet Protocol (ip), 20 bytes; Packets: 342 Displayed: 342 Marked: 0; Profile: Default.

Wireshark

- Detalles de un paquete

```
▶ Frame 329 (266 bytes on wire, 266 bytes captured)
▶ Ethernet II, Src: 02:00:00:00:01:f0 (02:00:00:00:01:f0), Dst: 02:00:00:00:02:f0 (02:00:00:00:02:f0)
▼ Internet Protocol, Src: 192.168.0.1 (192.168.0.1), Dst: 192.168.2.4 (192.168.2.4)
    Version: 4
    Header length: 20 bytes
    ▶ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 252
    Identification: 0xf1dd (61917)
    ▼ Flags: 0x00
        0.. = Reserved bit: Not Set
        .0. = Don't fragment: Not Set
        ..0 = More fragments: Not Set
    Fragment offset: 576
    Time to live: 64
    Protocol: ICMP (0x01)
    ▶ Header checksum: 0x0486 [correct]
    Source: 192.168.0.1 (192.168.0.1)
    Destination: 192.168.2.4 (192.168.2.4)
    ▶ [IP Fragments (808 bytes): #328(576), #329(232)]
▼ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0 ()
    Checksum: 0xc1c1 [correct]
    Identifier: 0x5d08
    Sequence number: 3 (0x0003)
    ▶ Data (800 bytes)
```