Universidad de Alcalá

Departamento de Automática

REDES DE COMPUTADORES
Laboratorio

Práctica 1: Emulación de redes con NetGUI.

Grado en Ingeniería Informática
Curso 2016/17

1. OBJETIVOS.

El objetivo de esta práctica es aprender a utilizar la herramienta de emulación de redes **Netkit / NetGUI**, y emplearla para adquirir y aplicar los conocimientos necesarios para la configuración y análisis de redes de datos basadas en TCP/IP.

2. ACTIVIDADES.

- Presentación y estudio de la aplicación **Netkit** y **NetGUI** para emular diferentes topologías de redes y poder realizar su configuración
- El alumno deberá estudiar con las aplicaciones **TCPdump** y **Wireshark** los resultados de las capturas de tráfico de red obtenidas a partir de las transferencias de datos realizadas.

3. Netkit y NetGUI.

3.1 Introducción: emulación de redes con Netkit y NetGUI.

Netkit es el resultado del trabajo conjunto del *Computer Network Research Group de la Universidad Roma Tre* y el grupo de usuarios de *Linux LUG Roma 3*. Se trata de un entorno software que permite realizar experimentos con redes de ordenadores virtuales sin necesidad de disponer de dispositivos de comunicaciones ni de ordenadores reales. Accedemos a ella a través de este enlace: http://wiki.netkit.org/

NetGUI ha sido desarrollada por el *Grupo de Sistemas y Comunicaciones* de la *Universidad Rey Juan Carlos*. Se trata de una interfaz gráfica para el sistema **Netkit** que permite editar diagramas de redes, arrancar las máquinas virtuales (ordenadores y routers) y pararlos, e interaccionar con las consolas de las máquinas virtuales. Esta herramienta de emulación de redes la utilizaremos para adquirir y aplicar los conocimientos necesarios en la configuración y análisis de redes de datos basadas en TCP/IP. Accedemos a ella a través de este enlace: http://mobiquo.gsyc.es/netgui/

3.2 Instalación de Netkit y NetGUI.

Se procederá a explicar como se ha de realizar la instalación manual sobre cualquier distribución de Linux (por si no se encontrase instalada en nuestro equipo), para ello seguiremos los siguientes puntos:

1. Es interesante instalar los siguientes paquetes software, en caso de no estar implementados en la distribución de Linux que se utilice:
   - firefox
   - xwit
   - telnetd
   - Java Runtime Environment 1.4.x o superior

2. Las instrucciones de instalación junto con los paquetes a descargar se encuentran en: http://mobiquo.gsyc.es/netgui/. Aquí disponemos de un fichero `INSTALACION.TXT` donde nos describe los pasos a realizar para proceder a la instalación dependiendo de la distribución de Linux con la que estemos trabajando.
3. Es imprescindible antes de empezar a trabajar con la aplicación de **NetGUI**, consultar el manual facilitado "`Pract_1 Introduccion_NetGUI.pdf` ".

3. 3 User Mode Linux (UML).

Netkit está basado en UML (*User-Mode-Linux*). UML permite arrancar una máquina Linux como un proceso de usuario corriendo dentro de una máquina anfitriona. Desde el punto de vista de la máquina anfitriona, UML es un proceso normal de usuario, véase figura 1. Desde el punto de vista de un proceso que corre dentro de UML, UML es un kernel, proporcionando memoria virtual y acceso a dispositivos, lo que denominamos máquina virtual.

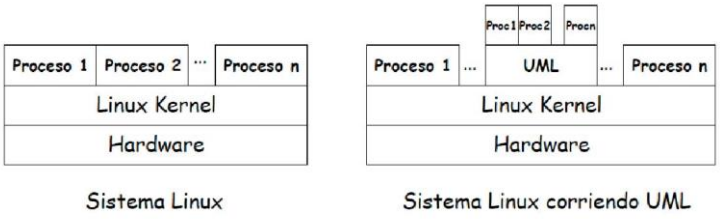

Fig. 1 - UML como un proceso de Linux

El kernel UML no se comunica directamente con el hardware, lo hace a través del kernel de Linux de la máquina anfitriona. Los procesos que corren dentro de UML funcionan igual que dentro de una máquina real Linux, ya que UML proporciona su propio espacio de direccionamiento de kernel y de proceso, su sistema de gestión de memoria y planificado, véase figura 2.

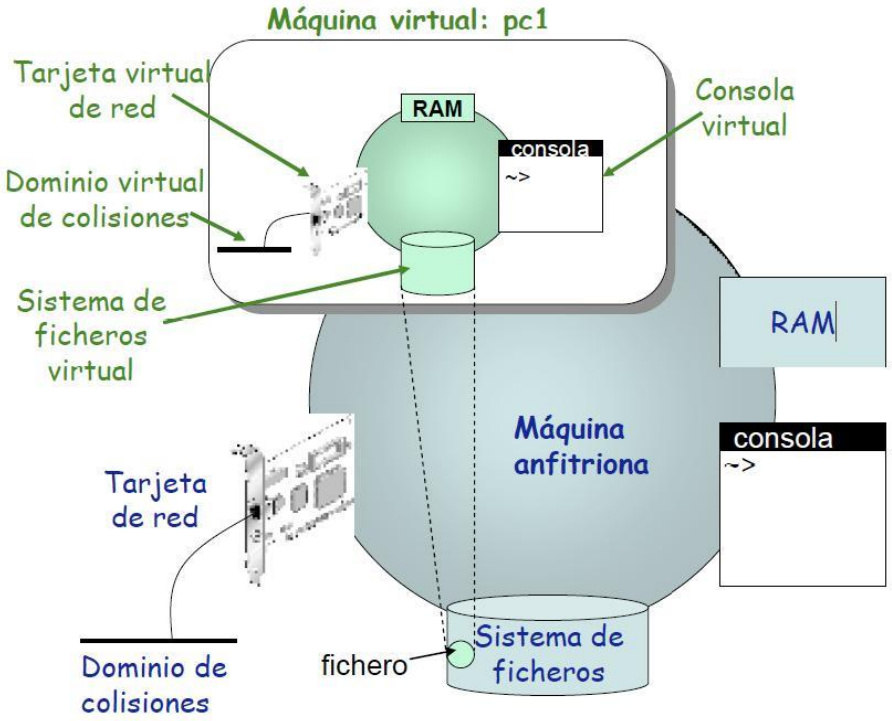

Fig. 2 Máquina virtual y máquina anfitriona

El sistema de ficheros que utiliza UML para arrancar cada máquina virtual se encuentra almacenado en un único fichero. Ahí es donde se encuentra el kernel de Linux y la configuración de una máquina virtual. Cuando queremos arrancar una máquina virtual, UML la arranca basándose en el kernel instalado en ese sistema de ficheros.

Gracias a UML podemos ejecutar un conjunto de máquinas virtuales dentro de una máquina real, la máquina anfitriona. Las máquinas virtuales se conectan a través de dominios de colisión virtuales (figura 3). Una máquina virtual puede funcionar como un equipo terminal, un router o un switch.

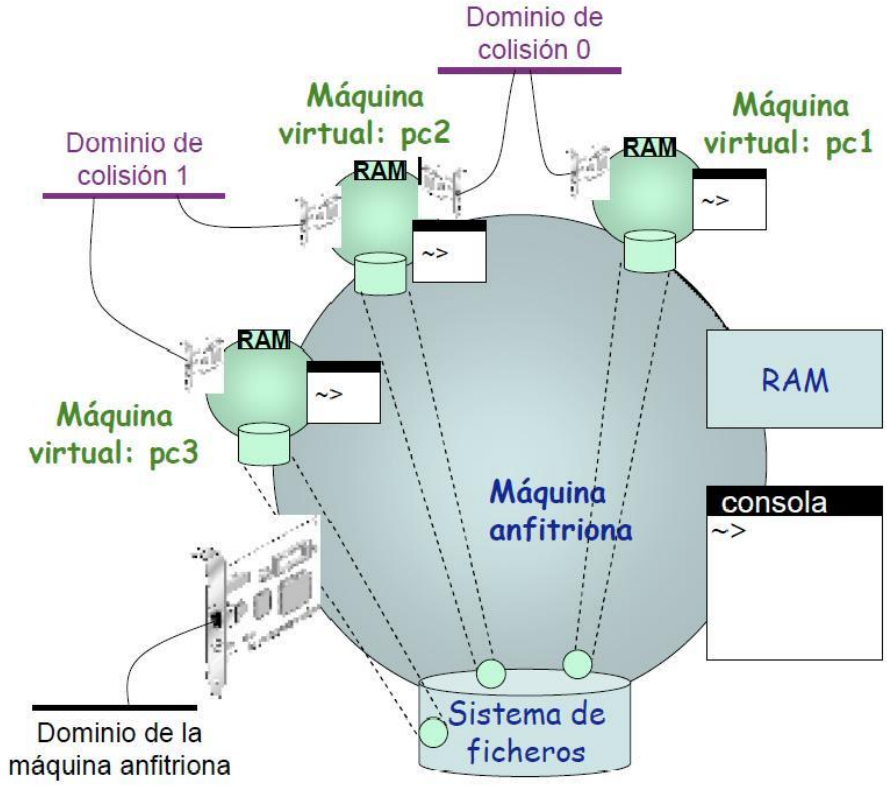

Fig. 3 - Conexión de máquinas virtuales a través de dominios de colisión

Valiéndose de la tecnología UML se desarrolló la herramienta **Netkit**, que es un conjunto de comandos que permiten configurar y conectar fácilmente máquinas virtuales, y un sistema de ficheros que contiene las herramientas necesarias para llevar a cabo la configuración en las máquinas virtuales. Y para que a la hora de trabajar con esta aplicación resultase más sencillo y ameno, se desarrollo herramienta **Netgui**, que es una interfaz gráfica que nos permite introducir los comandos de arranque y configuración necesarios para **Netkit** de una forma sencilla.

3.4 Configuración de una LAN.

Antes de continuar con los detalles de la práctica, se recomienda leer el manual de usuario que se ha facilitado de **NetGUI** para consultar el modo básico operación.

Arrancamos **NetGUI** y accedemos a las herramientas de edición de red. Estas herramientas nos permiten crear los siguientes elementos: *Terminal*, *Router*, *Hub* y *Conexiónes* entre dos de los elementos anteriores

Una vez dibujados los elementos que queremos emular, arrancaremos la configuración que nos abrirá las correspondientes consolas de configuración de los elementos que hayamos dibujado inicialmente. Crearemos varias redes y las analizaremos:

4. Realización de la Práctica

Los dos primeros ejercicios (1 y 2) te irán guiando en la utilización de Netgui, la configuración de equipos y los comandos de configuración y herramientas de análisis de redes.

Deberás realizar todos los ejercicios (1, 2 y 3) pero sólo **tendrás que entregar a tu profesor en un fichero PDF el ejercicio 3 (Responde a las cuestiones que aparecen en rojo)**

Ejercicio1:

Vamos a crear una red de dos ordenadores conectados directamente a través un *hub.*

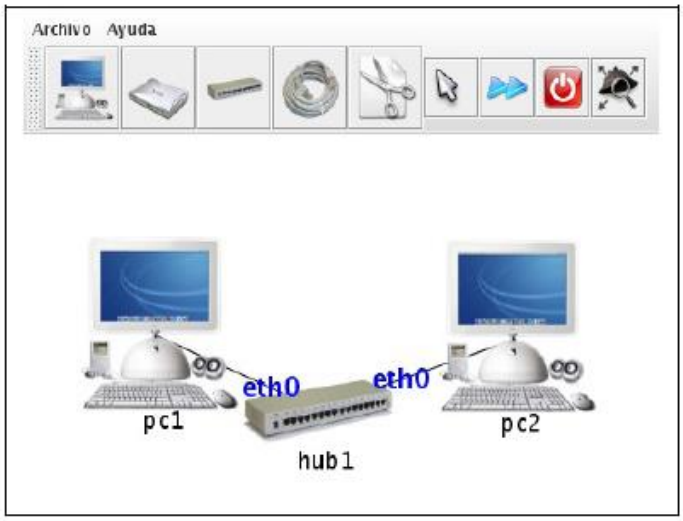

Fig. 4.

Para ello seguiremos los siguientes pasos:

1. Arrancar **NetGUI** (*netgui.sh*, en `/usr/local/netkit/netgui/bin`) y crea una red como la de la Figura 4. Guarda la configuración de la red con `Archivo→Guardar`. Elige como nombre de directorio **p0**, *(sin espacios).*

2. Arranca los ordenadores (*pc1* y *pc2*) y comprueba la configuración de red en cada uno de ellos con la orden *ifconfig.*

3. Las maquinas recién arrancadas no tienen configuradas sus interfaces de red Ethernet, por lo que ninguna aplicación podrá intercambiar mensajes con otras máquinas. Sólo podría hacerlo cada máquina consigo misma a través de la dirección 127.0.0.1, que está asignada a la interfaz de *loopback* lo. La interfaz de *loopback* es virtual, los datagramas que se envían a través de la interfaz de *loopback* vuelven a la misma máquina sin llegar a salir a la red.
   En pc1 ejecuta el comando `ping 127.0.0.1` Comprueba como la máquina **pc1** responde a sus propios *pings* destinados a la dirección 127.0.0.1. Repítelo para **pc2.**

4. Asigna 2 direcciones IP diferentes a cada máquina pero que ambas direcciones se encuentren dentro de la misma subred. Utiliza el comando *ifconfig* para asignar una dirección IP a una interfaz.

pc1 con: `$>ifconfig ethO 10.0.0.1 netmask 255.255.255.0`
pc2 con: `$>ifconfig ethO 10.0.0.2 netmask 255.255.255.0`
`(Automáticamente se generan las direcciones MAC de cada tarjeta de red)`

5. Comprueba que las máquinas son alcanzables entre sí. Desde **pc1** ejecuta el comando `ping 10.0.0.2`, y desde **pc2** ejecuta el comando `ping 10.0.0.1`. Eso significa que la red está bien configurada y funcionando correctamente.

6. Reinicia los equipos (*pc1 y pc2*) y comprueba si siguen conectados. Intenta razonar que ha pasado.

7. Para hacer cambios permanentes en las interfaces de los equipos es necesario modificar el fichero `/etc/network/interfaces`. Este archivo contiene información sobre las interfaces de red de un equipo y su configuración.

8. Antes de proceder a realizar estos cambios debemos levantar la interfaz. Para ello se pueden usar estos comandos:
   - `if up eth0`: levanta una interfaz y accede a la configuración del fichero `interfaces`.
   - `ifconfig eth0 up`: levanta una interfaz (se invoca indirectamente al configurar una interfaz con este comando). Si no se especifica configuración, la toma del fichero `interfaces`.

9. Modifica el archivo `interfaces` indicando las direcciones IP de cada equipo y 255.255.255.0 como máscara de subred. Para realizar estos cambios se puede utilizar el editor *vi*, siguiendo lo expuesto en la Pág. 27 del manual facilitado "`Pract_1 Introduccion_NetGUI.pdf`".

10. Reinicia los equipos, comprueba que las interfaces de red están bien configuradas y la red operativa *(ver paso 5).*

Vamos a analizar el tráfico que hay entre esas dos máquinas cuando realizamos un *ping*. Para ello vamos a utilizar la herramienta *tcpdump* que permite capturar todo el tráfico de una interfaz de red, y posteriormente lo analizaremos en la máquina anfitrión (real) con *wireshark.*

11. Arrancaremos *tcpdump* en una de las máquinas virtuales (**pc1)** con la opción *–l* para que la salida sea un buffer lineal y con la opción *–w* que permite guardar en un fichero todo el tráfico capturado.
    Ejemplo:
    pc1:~# `tcpdump –l –w /hosthome/pc1.eth0.cap`

12. Lanzaremos ping desde la otra máquina virtual para que se envíen 3 paquetes.
    Ejemplo:
    pc2:~# `ping –c3 10.0.0.1`

13. Cuando haya terminado *ping,* cerraremos *tcpdump* con *ctrl.+c*

14. Para analizar el fichero que ha escrito *tcpdump* tendremos que arrancar el programa *wireshark* en la máquina anfitriona, es decir, en la máquina real. Por tanto el fichero

donde está la captura hay que dejarlo en un directorio visible desde la máquina anfitriona, ese directorio es el directorio */hosthome* de la máquina virtual.
Dentro de una máquina virtual de NetGUI, al grabar cualquier fichero en el directorio */hosthome* permite guardar esos ficheros en la máquina real. *(Todos los ficheros grabados en el directorio* ***/hosthome*** *estarán en realidad en el* ***$HOME*** *del usuario en la máquina real)*

15. Puede arrancarse *wireshark* desde un terminal de la máquina real de la siguiente forma:
    Ejemplo:
    usuario@xxx:~$ `wireshark pc1.eth0.cap`

Ejercicio2:

Crea una red como la de la Figura 5, en donde **pc1, pc2** son 2 *ordenadores*, y **r1** es un *router*. Para conectar varios dispositivos a un mismo segmento de red necesitarás utilizar *hubs*:

- *hub1* para conectar *pc1* y *r1*.
- *hub2* para conectar *pc2* y *r1*.

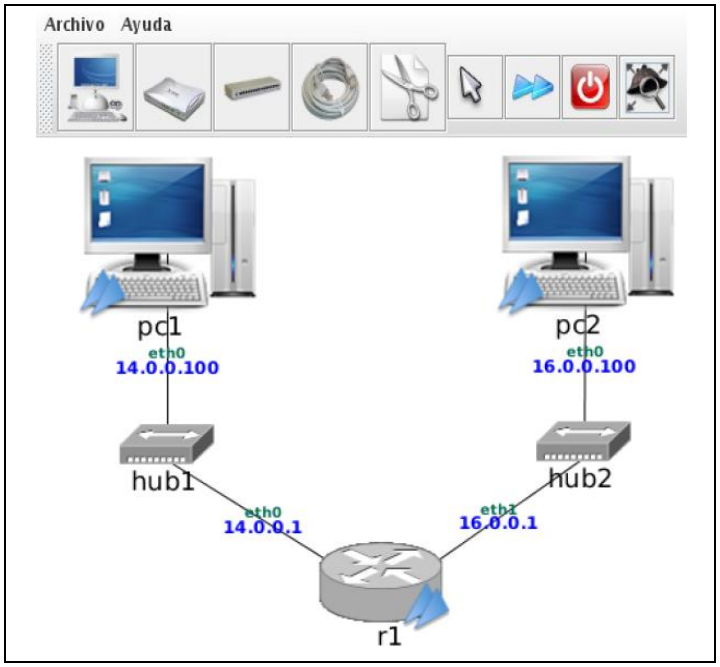

Fig. 5.

1. Arrancar **NetGUI** (*netgui.sh*, en `/usr/local/netkit/netgui/bin`) y crea una red como la de la Figura 5. Guarda la configuración de la red con `Archivo→Guardar`. Elige como nombre de directorio **p1**, *(sin espacios).*
   Fíjate en el orden en el que dibujas los cables para que las interfaces de los routers se enumeren de la misma forma que aparecen en el dibujo. En el caso de **r1** primero deberás dibujar el cable que une a **r1** con **hub1** y después el que une **r1** con **hub2**. De esta forma las interfaces eth0 y eth1 quedarán enumeradas de la misma manera que en el dibujo. (*en NetGUI las direcciones IP no aparecen coloreadas*)

2. Arranca los ordenadores y el encaminador de uno en uno, y comprueba la configuración de la red en cada uno de ellos mediante la orden *ifconfig.*

3. Los ordenadores sólo tienen activa la interfaz de *loopback* (lo) con la dirección 127.0.0.1. Comprueba que la interfaz de *loopback* está operativa con el comando `ping 127.0.0.1` en los dos ordenadores. Los equipos tienen otra interfaz de red (*ethO*), pero que no está activada

4. Comprueba con la orden *route* la tabla de encaminamiento de las máquinas. Verás que está vacía en todas las máquinas

5. Utilizando las órdenes *ifconfig* o *ip*, asigna las siguientes direcciones IP a las interfaces de red de las máquinas de la siguiente forma:

    - `pc1 (eth0): 14.0.0.100`
    - `pc2 (eth0): 16.0.0.100`
    - `r1 (eth0): 14.0.0.1`
    - `r1 (eth1): 16.0.0.1`
    - La máscara de subred debe ser en todos los casos la 255.255.255.0
    - Comprueba que cada interfaz tiene la dirección IP adecuada llamando a *ifconfig* sin argumentos en cada máquina.

6. Con el comando *route* puedes comprobar como tras asignar la dirección IP a una interfaz de red de una máquina se añade automáticamente una entrada en la tabla de encaminamiento de la máquina. Ahora deberá aparecer una entrada que permite encaminar datagramas hacia cualquier dirección de la misma subred a la que nos conecta la interfaz.

7. Comprueba como ahora sí funcionan los *ping* a direcciones de la misma red. Haz `ping 14.0.0.1` desde **pc1** y `ping 16.0.0.1` desde **pc2**.

8. Comprueba como al hacer desde **pc1** un `ping 16.0.0.1`, no obtenemos resultados (no funciona).

9. Añade una ruta por defecto en **pc1** para que los datagramas IP que no sean para su propia red los envíe a 14.0.0.1. Comprueba que ahora el ping a 16.0.0.1 desde pc1 sí funciona.

10. Sin embargo, sigue sin funcionar el ping de **pc1** a **pc2**. Intenta razonar:¿Por qué? Soluciónalo y comprueba que funciona el ping de **pc1** a **pc2**. (*) Solución en Apéndice

11. Si en el punto 5 utilizaste ordenes en la ventana de terminal, las direcciones asignadas no se conservarán si apagas las máquinas y las vuelves a encender. Edita ahora el fichero `/etc/network/interfaces` de todas las máquinas de forma que se pueda hacer ping desde cualquier máquina a cualquiera de las direcciones IP del dibujo. Recuerda que para que se efectúen las modificaciones realizadas sobre ese fichero es necesario reiniciar las interfaces de red con la orden `/etc/init.d/networking restart`, después de haber editado el fichero `interfaces`.

12. Apaga las máquinas de una en una. Cuando lo hayas hecho, mira en la máquina real los ficheros que hay en el directorio **p1**. Verás un fichero *netgui.nkp*, que contiene

una descripción del dibujo del escenario de red creado. Además, verás un fichero de extensión *.disk* por cada máquina del escenario. En cada fichero *.disk* están los cambios realizados sobre el sistema de ficheros original de una máquina nueva recién creada en NetGUI. Así, por ejemplo, en *pc1.disk* estarán los cambios realizados sobre el `/etc/network/interfaces` de esa máquina. Estos ficheros *.disk* son ficheros "dispersos" (*sparse files*). Estos ficheros tienen la característica de tener muchos bytes a cero que, por razones de ahorro espacio, no están realmente presentes en el disco mientras sigan estando a cero. A veces se dice que son ficheros "con huecos". Por eso estos ficheros parecen muy grandes (600 MB) cuando se ve su tamaño con *ls -l* , pero al ver lo que ocupan realmente en disco con *du -s* , se ve que su tamaño real es mucho menor (600 KB). Estos ficheros no se pueden transportar fácilmente de un dispositivo a otro (por ejemplo, a una memoria usb) o de un ordenador a otro (por ejemplo, de casa al laboratorio), puede ser que en uno de estos intercambios se decida convertir los huecos en bytes reales y pasan a ocupar en disco los 600 MB cada uno.

13. Vuelve a encender las máquinas (de una en una) y comprueba que se han mantenido las direcciones IP y las rutas igual que antes de apagar. Comprueba que puedes hacer ping entre todas las máquinas y entre las máquinas y el router.

14. Utilizando las herramientas *tcpdump* y *wireshark*, intenta razonar sobre siguientes cuestiones planteadas respecto al escenario anterior. Para ello deberás decidir en qué momento y en qué subredes tienes que arrancar *tcpdump*. Haz un *ping* de **pc1** a **pc2** para que envíe 3 paquetes ICMP: (*) Solución en Apéndice
    - Con qué TTL llegan los mensajes ICMP a **pc2**?
    - Qué valor tienen los campos *Type* y *Code* de los mensajes ICMP que llegan a **pc2**?
    - Qué valor tienen los campos *Type* y *Code* de los mensajes ICMP que llegan a **pc1**?
    - Qué valor numérico hexadecimal tiene el campo Protocolo de los datagramas IP en los que viajan los mensajes ICMP?
    - Qué valor numérico hexadecimal tiene el campo de tipo de protocolo de las tramas Ethernet en las que viajan los mensajes ICMP?
    - En qué subred/subredes has tenido que capturar el tráfico con *tcpdump* para responder a las preguntas de este apartado?

Apéndice: Solución a los problemas planteados en el Ejercicio 2.

10. No funciona el *ping* de **pc1** a **pc2** porque, aunque ya hay ruta en **pc1** para alcanzar a **pc2, pc2** no tiene ruta para alcanzar **pc1**. Para que funcione el *ping* deben poder enviarse datagramas IP en ambos sentidos. Sí funciona desde **pc1** el *ping* a la 16.0.0.1 porque esa IP es de **r1**, y **r1** sí tiene ruta para alcanzar a **pc1**. Para solucionarlos, hay que añadir en **pc2** una ruta para alcanzar **pc1**, o la red de **pc1**, o una ruta por defecto, a través de **r1**.

14. Las respuestas que se deben obtener, se han de parecer a:
    - Los ICMP *echo request* llegan a **pc2** con TTL 63, ya que salen de **pc1** con valor 64, y se decrementan en 1 al pasar por **r1**.
    - Los ICMP *echo request* que llegan a **pc2** tienen *Type* 8 y *Code* 0.
    - Los ICMP *echo reply* que llegan a **pc1** tienen *Type* 0 y *Code* 0.

- Los datagramas IP que contienen mensajes ICMP tienen en su campo *Protocol* el valor 0x01.
- Las tramas Ethernet que contienen datagramas IP tienen en su campo (*Protocol*) *Type* el valor 0x0800.
- Para ver los paquetes que llegan a **pc2** puede capturarse en **pc2**, interfaz eth0. Para ver los paquetes que llegan a **pc1** puede capturarse en **r1**, interfaz eth0.

Ejercicio3:

Crea una red como la de la Figura 6, en donde **pc1, pc2, pc3** son 3 *ordenadores*, **r1** es un *router*, y **hub1, hub2** son *hubs*.

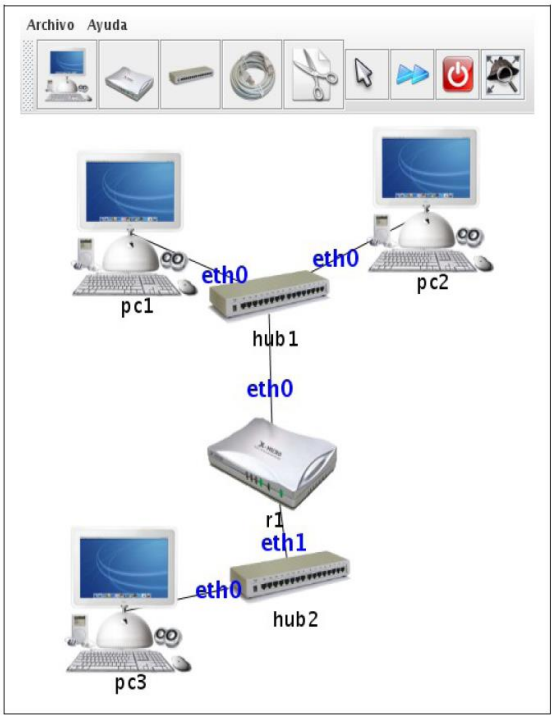

Fig. 6.

1. Arrancar **NetGUI** (*netgui.sh*, en `/usr/local/netkit/netgui/bin`) y crea una red como la de la Figura 6. Guarda la configuración de la red con `Archivo→Guardar`. Elige como nombre de directorio **p2**, *(sin espacios).*
   Fíjate en el orden en el que dibujas los cables para que las interfaces de los routers se enumeren de la misma forma que aparecen en el dibujo

2. Arranca los ordenadores y el encaminador de uno en uno, y comprueba la configuración de la red en cada uno de ellos mediante la orden *ifconfig.*

3. Utilizando las órdenes *ifconfig* o *ip*, asigna las siguientes direcciones IP a las interfaces de red de las máquinas de la siguiente forma:
   - `pc1 (eth0): 192.168.1.10`
   - `pc2 (eth0): 192.168.1.11`
   - `pc3 (eth0): 192.168.2.10`
   - La máscara de subred debe ser en todos los casos la 255.255.255.0

4. El router r1 tiene dos interfaces de red: eth0 en la red 192.168.1.0 y eth1 en la red 192.168.2.0. Por norma, el router de una red suele tener asignada la primera dirección IP del rango. Configura las interfaces de red de **r1**:
   - eth0 192.168.1.1
   - eth1 192.168.2.1

5. Comprueba que desde **pc3** se alcanza la dirección 192.168.2.1
   - Ejecuta: `$> ping 192.168.2.1`

6. Comprueba que desde **pc1** y **pc2** se alcanza la dirección 192.168.1.1
   - Ejecuta: `$> ping 192.168.1.1`

7. Comprueba si desde **pc1** y **pc2** se alcanza la dirección 192.168.2.10
   - Ejecuta: `$> ping 192.168.2.10`

Intenta razonar los resultados de los anteriores apdos **5, 6 y 7**:¿Qué está pasando?

8. Hay que indicar a cada equipo cuál es su router por defecto. Configura el fichero `interfaces` de **pc1, pc2** y **pc3** para que utilicen r1 como su router por defecto.

Entrega un pantallazo de la configuración

9. Comprueba que ahora es posible desde **pc1** (*o **pc2***) alcanzar a **pc3** (y viceversa).

10. Comprueba cómo queda la tabla de enrutamiento de **pc1, pc3** y **r1**, con el comando *route. (También es posible modificar la tabla de enrutamiento con el comando **route**, aunque para que el cambio sea permanente es preferible utilizar el fichero `interfaces`).*

ANALISIS DE TRÁFICO

Vamos a analizar el tráfico que hay entre esas tres máquinas cuando realizamos un *ping*. Para ello vamos a utilizar la herramienta *tcpdump* que permite capturar todo el tráfico de una interfaz de red, y posteriormente lo analizaremos en la máquina anfitrión (real) con *wireshark*.

11. En primer lugar, vamos a ***capturar y visualizar*** el tráfico que circula a través de **r1** por la red 192.168.2.0, con el comando *tcpdump.* (podemos también realizar las modificaciones de ejecución oportunas que nos permita visualizar y estudiar los resultados a través de *wireshark*)
    - Ejecuta: `$> tcpdump -i eth1 -s 0 -w /hosthome/traffic1.cap`

12. En **pc3**, vamos a comprobar que la cache ARP está vacía con el comando *arp*. En caso de no estar vacía, con la opción `-d <host>` se pueden eliminar las entradas existentes.

13. Desde **pc3** realiza lo siguiente:
    - Ejecuta: `$> ping 192.168.1.10`.
    - Intenta razonar:¿De qué tipo es el primer mensaje que captura **r1**? ¿Y los siguientes?

14. Seguimos en **pc3**, y vamos a ampliar el campo de datos en los mensajes *ping* con la opción `-s`:
    - Ejecuta: `$> ping -s 2000 192.168.1.10`
    - Intenta razonar:¿Cuántos mensajes se capturan en **r1** por cada ICMP *echo request*?

15. Para comprobar el tamaño de los mensajes que se envían es necesario capturarlos (en **r1**) con el siguiente comando:
    - Ejecuta: `$> tcpdump -i eth1 -e -w /hosthome/traffic5.cap`
    - Repite el paso 14 y comprueba el tamaño de los mensajes enviados e intenta razonar: ¿Cuántos bytes de datos se envían en el primer datagrama? ¿Y en el segundo?.

16. Para enviar datos mediante UDP y TCP se utiliza el comando `nc`.
    - Lanzar un servidor en **pc1** en el puerto 1111:
      Ejecuta: `$> nc -l -p 1111`
    - Lanzar un cliente en **pc3** para que se conecte con el servidor:
      Ejecuta: `$> nc 192.168.1.10 1111`
    - Intenta razonar:
      - ¿Qué mensajes se han capturado en **r1**?
      - Teclea algo en el cliente y pulsa ENTER. ¿Qué circula por la red?
      - Para cerrar la conexión pulsa *Ctrl-C* en el cliente. ¿Cuántos mensajes se intercambian para cerrar la conexión?

17. Vamos a comprobar qué sucede cuando intentamos establecer una conexión a un puerto no existente. (Arranca de nuevo el servidor)
    - Lanzar un cliente en **pc3 (sin lanzar el servidor)**
      Ejecuta: `$> nc 192.168.1.10 1111`
    - Intenta razonar: ¿Qué tipo de mensaje envía el servidor?

18. Repetimos el paso 6 pero ahora con UDP:
    - Lanzar un servidor en **pc1** en el puerto 1111:
      Ejecuta: `$> nc -l -u -p 1111`
    - Lanzar un cliente en **pc3** para que se conecte con el servidor:
      Ejecuta: `$> nc -u 192.168.1.10 1111` (pulsa enter, teclea unos datos y vuelve a pulsar enter).
    - Intenta razonar: ¿Qué respuesta obtenemos del servidor? ¿Cuántos mensajes se intercambian para cerrar la conexión?
    - ¿Qué diferencias encuentras entre UDP y TCP?

5. Bibliografía.

- La señalada como básica para el laboratorio.
- **NetGUI** ha sido desarrollada por el *Grupo de Sistemas y Comunicaciones* de la *Universidad Rey Juan Carlos*. (Madrid, España).
  http://mobiquo.gsyc.es/netgui/
- **Netkit** es el resultado del trabajo conjunto del *Computer Network Research Group* de *Universidad Roma Tre* y el grupo de usuarios de *Linux LUG Roma 3*.
  http://wiki.netkit.org/

- Para esta práctica se ha tomado como base el Manual de Prácticas de la asignatura de *Redes I* del Departamento de Sistemas Telemáticos y Computación (GSyC) de la *Universidad Rey Juan Calos* (Madrid, España).

6. Material adicional

- **Tcpdump**
  - Manual oficial.
    http://www.tcpdump.org/tcpdump_man.html
  - Manual Reference.
    http://www.gsp.com/cgi-bin/man.cgi?section=1&topic=tcpdump