Máquinas Virtuales en las clases de Informática

Pedro Pablo Gómez Martín
Grupo de Aplicaciones de Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid
pedrop@fdi.ucm.es

Resumen

La enseñanza de algunas asignaturas del currículo de Informática se verían enriquecidas con la realización de prácticas para las que los alumnos necesitan privilegios de administración. Esto está reñido con el control que los administradores de los laboratorios imponen sobre sus ordenadores, para conseguir un funcionamiento correcto de forma ininterrumpida. Este artículo propone el uso de máquinas virtuales para romper esta paradoja.

1. Introducción

La enseñanza de muchas asignaturas relacionadas con Informática no puede darse completamente por terminada si no se enriquece con clases prácticas. Así, hoy en día nadie plantea una asignatura de programación sin ejercicios en el laboratorio.

Por desgracia, algunas asignaturas, sin embargo, siguen quedando descompensadas, al impartirse con una alta carga teórica, apoyada por una escasa, incluso inexistente, ayuda práctica.

Esta carencia no siempre es achacable al profesor. En determinadas asignaturas las dificultades aparecen debido a cuestiones organizativas. Habitualmente, los laboratorios son utilizados para diferentes tareas simultáneamente. No es extraño que a primera hora de la mañana un ordenador se utilice para realizar prácticas de programación en C, dos horas después para bases de datos, y termine el día ejecutando código en Prolog.

El resultado es que el buen funcionamiento de los ordenadores de los laboratorios es *crítico* para la impartición de muchas asignaturas. Por ello, los técnicos que los mantienen no quieren verse en la desagradable situación de que de una sesión a otra dejen de funcionar varios ordenadores. Aunque una solución son las herramientas de clonación, a menudo terminan restringiendo drásticamente los privilegios de los alumnos en los ordenadores. Les impiden instalar software, limitan su uso de disco e incluso les prohíben personalizar el escritorio.

Todas estas restricciones están completamente justificadas desde el punto de vista organizativo y de *imagen*: no sería razonable que al encender un ordenador apareciera un fondo o protector de pantalla de dudoso contenido, o que los laboratorios de una Facultad fueran un hervidero de virus.

Sin embargo, también impiden realizar actividades que pueden resultar pedagógicamente importantes. Por ejemplo, los alumnos que aprueban la asignatura de bases de datos consiguen conocimientos sobre su diseño, normalización, SQL y, quizá, algún gestor de bases de datos como Oracle o MySQL. En este último caso, sin embargo, rara vez se llega al *nivel físico* de la base de datos, porque los alumnos deberían tener privilegios sobre ella, algo que suele considerarse *peligroso*.

Del mismo modo, en la asignatura de Sistemas Operativos los alumnos no suelen ni siquiera crear un usuario en el sistema, porque para eso tendrían que ser sus administradores. Obviamente, las restricciones no son un problema para muchas otras asignaturas; pero la imposibilidad de realizar ciertas prácticas en algunas de ellas es una lacra con la que durante años se ha tenido que convivir.

Este artículo propone una solución a este problema. Para eso, describe de forma general el funcionamiento del software de virtualización y emulación y enumera las herramientas existentes. Luego plantea los posibles usos de esta tecnología en las clases prácticas de Informática, detalla y evalúa un ejemplo concreto, y finaliza con el trabajo futuro y unas conclusiones.

2. Software de virtualización y emulación

Durante los primeros años de la informática, los grandes ordenadores eran utilizados únicamente por sus diseñadores. Éstos hacían un uso "interactivo" de sus sistemas, accediendo a ellos directamente, y vigilándolos por si surgía algún problema. En esos días, los ordenadores se utilizaban para realizar cálculos masivos que requerían mucho tiempo, debido a su, retrospectivamente

hablando, baja velocidad. Como consecuencia de ello, la interacción hombre-máquina no suponía una excesiva pérdida de recursos, pues el tiempo de manipulación era despreciable si se comparaba con el tiempo de ejecución de cada programa.

Durante la década de los 60 los ordenadores se hacen mucho más rápidos, y el tiempo dedicado a la entrada/salida deja de ser despreciable. El funcionamiento por lotes se generaliza, permitiendo un mayor aprovechamiento de la CPU a costa de un uso mucho más indirecto de los sistemas.

Afortunadamente, tiempo después aparecerían los sistemas operativos de *tiempo compartido*, que recuperaban el uso original interactivo permitiendo, además, un nivel de utilización alto de la CPU.

Precisamente fruto del trabajo en sistemas operativos durante la década de los 60, en 1972 IBM publica su Virtual Machine Facility/370 (abreviada como VM/370), la arquitectura software para su nueva familia de *mainframes* IBM System/370, y que sirvió para acuñar por primera vez el término *máquinas virtuales* [1]. La idea que propone es ejecutar sobre el *hardware* un pequeño sistema operativo que *simula* al propio hardware subyacente un número de veces arbitrario. Cada una de las máquinas simuladas es una copia exacta del *hardware* real, pero recreado de forma lógica. El software que hace esto realidad fue denominado CP (*Control Program*).

Sobre las máquinas lógicas se ejecuta un *sistema operativo completo*, que tiene la ilusión de estar corriendo sobre la máquina desnuda. De hecho, es posible ejecutar en cada una de ellas un sistema operativo diferente aunque, en la práctica, IBM impulsaba el uso de CMS (*Conversational Monitor System*).

CP proporcionaba así múltiples *entornos de computación*, permitiendo ejecutar un sistema operativo diferente sobre cada uno de ellos. Éstos tenían la sensación de ejecutarse en modo supervisor, dando apoyo a las aplicaciones normales en modo usuario. Sólo cuando el sistema operativo (habitualmente CMS) accedía al *hardware*, CP entraba en acción para crear la simulación y evitar las interacciones no deseadas entre máquinas.

El tercer componente de la arquitectura VM/370 lo componía RSCS (*Remote Spooling and Communications Subsystem*), encargado de conseguir que las diferentes máquinas virtuales pudieran comunicarse entre sí.

Con esta arquitectura por capas, IBM consiguió un sistema de tiempo compartido. De hecho, la sensación de control de los usuarios era inmensa, pues CMS era en realidad un sistema operativo *monousuario*, pero que al ser ejecutado en múltiples instancias gracias a CP permitía un mayor aprovechamiento del *hardware*.

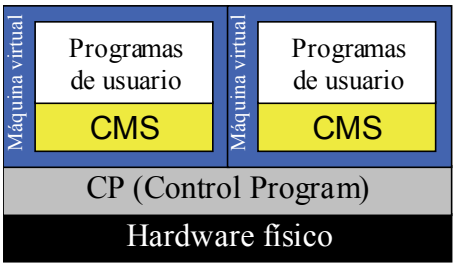

Figura 1. Arquitectura de VM/370

A pesar de su brillantez, esta idea no se generalizó. Los sistemas operativos de tiempo compartido "reales", como Multics y, especialmente Unix, terminaron adueñándose del mercado, y en la década de 1.980 las máquinas virtuales perdieron fuerza.

Con el paso del tiempo, los ordenadores se hacen más potentes y surge la posibilidad de *emular* máquinas antiguas en los ordenadores nuevos. Quizá los emuladores más conocidos sean los de las máquinas de 8 bits con las que, debido al "gancho" de sus juegos, muchos se iniciaron en el mundo de la Informática. Surgen así distintos emuladores de las máquinas Amstrad, Spectrum o Commodore. Las máquinas recreativas y las consolas no han escapado a esta ola de nostalgia, como demuestra la gran aceptación de Mame [2]. En ámbitos menos lúdicos también existen emuladores de computadores míticos como Hercules [3] que emula algunos mainframes de IBM.

La aproximación de los emuladores es diferente a la idea de las máquinas virtuales. Un emulador se ejecuta sobre un sistema operativo normal, y consiste en un *intérprete software* de las instrucciones en código máquina de la CPU que se simula. Naturalmente, como en el caso de las máquinas virtuales también se simula el resto del *hardware*, con los dispositivos de E/S al frente.

La diferencia entre ambas posibilidades marca sus características opuestas. En el caso de los emuladores es posible construir software para simular cualquier arquitectura *hardware*, siempre

que se disponga de suficiente capacidad de cálculo[1]. Por su parte, la aproximación que utiliza máquinas virtuales únicamente puede *clonar* la máquina original sobre la que se ejecuta, recreando quizá *hardware* auxiliar ligeramente diferente pero siempre sobre la misma CPU (mismo código máquina) que la original.

Con la mejora de las prestaciones de los ordenadores de escritorio y servidores, la idea de la virtualización del *hardware* que nació con VM/370 ha recobrado interés en el mundo PC. Así, han surgido algunas aplicaciones interesantes que crean varios "PCs lógicos" dentro de uno solo, permitiendo ejecutar simultáneamente varios sistemas operativos. También existen varios *emuladores* de PC que se ejecutan tanto en arquitecturas PC como en otras. Algunos, sin embargo, cuando detectan que la máquina subyacente está compuesta por la CPU que tratan de emular, ante la menor oportunidad pasan a ejecutar directamente el código en lugar de utilizar emulación. Así el rendimiento mejora, y se difumina la diferencia entre software de virtualización y emulador.

En el resto del artículo nosotros nos interesaremos precisamente por las herramientas que consiguen recrear arquitecturas PC, ya sea mediante emulación o virtualización. Desde el punto de vista del usuario, hoy en día ambas son indistinguibles en su uso, pues, salvo por alguna excepción, requieren la existencia de un sistema operativo *completo* al que se le denomina anfitrión (o "host"). Sobre él se ejecuta la aplicación que construye la máquina virtual, y que permite la ejecución de otro sistema operativo *invitado*.

La máquina virtual recrea *todo* el sistema PC. Para configurarla, hay que especificar el fichero (en el sistema operativo anfitrión) que contiene el "disco duro" tal y como lo verá el sistema operativo invitado. Éste no será consciente, naturalmente, de que esta utilizando un fichero; en lugar de eso accederá al "hardware" del disco del modo tradicional, con sectores, cilindros y cabezas. También se configura el modo en el que se simula la tarjeta de red, la tarjeta de sonido, e incluso el teclado.

[1] Se estima que para simular una arquitectura determinada se necesita un sistema que disponga alrededor de 10 veces la capacidad de cálculo que el sistema emulado.

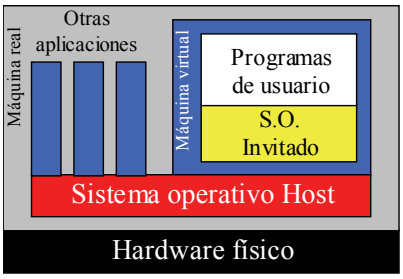

Figura 2. Máquina virtual sobre un anfitrión

La simulación es tal que se realizan las comprobaciones de la BIOS pudiendo a veces, incluso, arrancar directamente desde la red gracias al estándar PXE (Preboot Execution Environment, [4]) aunque no esté disponible en la BIOS del ordenador anfitrión.

Normalmente el primer arranque fallará, como pasaría en cualquier ordenador real por la ausencia de sistema operativo en el "disco". Para instalarlo, basta con meter el CD autoarrancable correspondiente en el ordenador real. El simulador lo detecta, y hace creer a la BIOS del ordenador simulado que dispone de un lector de CD con el instalador.

Naturalmente todo esto hace patente la necesidad de un espacio de almacenamiento en disco (físico) considerable, para poder así mantener los ficheros asociados a los discos lógicos de las máquinas virtuales. Además, el uso de memoria RAM se dispara, porque cada máquina virtual debe reservar tanta como memoria física asegure que tiene disponible.

En las siguientes secciones se realiza una descripción de las alternativas más representativas para conseguir máquinas virtuales sobre un PC.

2.1. Bochs

Bochs es quizá el *emulador* de la arquitectura PC más conocido en el entorno del software libre [5]. Desarrollado en C++ originalmente bajo una licencia propietaria, Mandrake (hoy conocido como Mandriva) adquirió los derechos y liberó su código bajo licencia LGPL. Su configuración es relativamente compleja, dada su amplia capacidad de personalización. Está disponible para una gran cantidad de plataformas, incluyendo Windows, Linux y MacOS X.

Figura 3. Windows 2000 en Bochs sobre Linux

2.2. QEmu

QEmu [6] es la alternativa a Bochs, también distribuida bajo licencia libre para *emular* la arquitectura PC. Su configuración es mucho más sencilla, lo que le ha hecho ganar adeptos con el tiempo. Permite también emular otras arquitecturas, como PowerPC.

Su modo de funcionamiento difiere a la de Bochs porque QEmu es en realidad un *traductor dinámico de instrucciones*. Las instrucciones que la CPU simulada se va encontrando se traducen a la máquina nativa para una ejecución más rápida.

QEmu estaba disponible originalmente para GNU/Linux, aunque han aparecido versiones para Windows, FreeBSD y Mac OS X con diferente grado de fiabilidad. Cuando se ejecuta sobre Linux o Windows en arquitectura PC, es posible cargar un *módulo* específico para convertir a QEmu en software de virtualización. Este módulo, sin embargo, no se distribuye con una licencia de código abierto, aunque su uso es gratuito.

2.3. Virtual PC

Virtual PC [7] es la alternativa de Microsoft para la *emulación* de la arquitectura PC. Originalmente desarrollado por Connectix, fue adquirido en 2003 con la intención de ofertar una solución software a los clientes que deseaban actualizar sus sistemas operativos a versiones más modernas y aun así querían ejecutar aplicaciones heredadas que sólo funcionan en sistemas operativos antiguos. También está disponible para Mac, permitiendo ejecutar Windows en la arquitectura de la competencia.

2.4. VMWare

VMWare [8] es quizá el software de *virtualización* de PC más conocido. La empresa surgió en 1998, formada por parte de los integrantes de un grupo dedicado a la investigación en el ámbito de los Sistemas Operativos en la Universidad de Stanford. Se ejecuta sobre Windows y sobre GNU/Linux como anfitriones, y permite la ejecución sin cambios de gran cantidad de sistemas operativos *huesped*. Como no utiliza emulación, proporciona un buen rendimiento. Además, permite instalar controladores en algunos de los posibles sistemas operativos invitados para adaptarlos al *hardware* que VMWare está simulando por debajo. Eso aumenta aún más el rendimiento y facilita su utilización al integrarlo mejor con el sistema anfitrión (especialmente en lo referente al ratón).

Aparte de la facilidad de instalación, uso y la estabilidad que un software comercial maduro puede proporcionar, permite la simulación de *segmentos de red virtuales*, para unir máquinas virtuales ejecutadas en un mismo anfitrión a través de una conexión *lógica* por una Ethernet simulada. Salvando las distancias, esto se podría comparar con el subsistema RSCS de VM/370 para comunicar diferentes máquinas virtuales.

También existe una versión de VMWare pensada para servidores que *no necesita un sistema operativo anfitrión*. En lugar de eso, se ejecuta directamente sobre el metal desnudo de la arquitectura subyacente, tal y como ya hiciera el CP (Control Program) de VM/370 décadas atrás.

2.5. Otras alternativas

Hoy en día, el uso más claro para la virtualización y emulación se centra en servidores. Las empresas que proporcionan servicios de *hosting* para clientes "poco exigentes" en recursos perderían mucho dinero proporcionando a cada uno un servidor completo que va a estar ocioso el 95% del tiempo. En lugar de eso, se simulan varios servidores dentro de un único ordenador físico. El cliente sigue teniendo la ilusión de tener un servidor completo para él (con el sistema operativo de su elección), y aun así la empresa ahorra dinero en *hardware* y en consumo energético.

Aparte de las descritas, existen muchas otras herramientas que consiguen este efecto de forma muy eficiente, a costa de estar *atadas* a un determinado sistema operativo anfitrión e invitado.

Quizá donde más alternativas hay es para la creación de varios servidores GNU/Linux sobre un ordenador físico corriendo una versión "*nativa*" también de GNU/Linux. Ejemplos de este tipo de herramientas son User Model Linux (UML) o Linux-VServer. A menudo se las conoce como "jail systems", porque amplían el concepto del comando `chroot` del mundo de Unix hasta englobar a todo el sistema (incluido el núcleo).

3. Posibles usos del software de virtualización en las clases de Informática

Salvo en el último caso, cualquiera del resto de aplicaciones anteriores consigue, como se ha dicho, simular un PC completo sobre un ordenador anfitrión que nunca se verá afectado por los problemas que aparezcan en aquél. Esto es de aplicación directa en los laboratorios de Informática, dado que los alumnos pueden utilizar máquinas virtuales sin restricciones, y aun así los administradores de los laboratorios podrán estar seguros de que los ordenadores físicos no sufrirán daños y estarán disponibles para la siguiente clase. En un determinado momento un alumno podría dejar inutilizable *su* máquina virtual, pero al no afectar al sistema operativo anfitrión, no surgirán problemas.

La propuesta para el uso de este tipo de software *no* es proporcionar una máquina virtual para cada alumno y que la utilicen *siempre*. Las máquinas virtuales, con su mayor consumo en memoria y en disco, se usarían *únicamente* en aquellas asignaturas en las que permitir a los alumnos experimentar con derechos de administrador tenga valor pedagógico. Para el resto, se seguirá utilizando el ordenador real como se venía haciendo tradicionalmente. Además, la existencia de las máquinas virtuales no impone una carga adicional de trabajo para los administradores de los laboratorios (más allá de la instalación del software), dado que ellos *no* serán los encargados de conseguir el correcto funcionamiento de dichas máquinas, sino que serán responsabilidad de los alumnos que las administren como práctica para la asignatura correspondiente.

El objetivo, por lo tanto, es que cada máquina virtual sea utilizada por un alumno (o por un grupo de ellos) para una *única asignatura* que lo requiera. Si cometen algún error y queda inutilizable, aún podrán realizar sus prácticas de otras asignaturas sobre la máquina real, o sobre una máquina virtual diferente.

Como se ha comentado, las asignaturas candidatas a utilizar este tipo de software en sus prácticas son aquellas en las que puede resultar de interés que los alumnos tengan privilegios de administración en algún sentido. Tal y como se mencionaba en la introducción, un posible ejemplo es en la asignatura de bases de datos. Si, además de los aspectos relacionados con el *diseño* de bases de datos y la realización de consultas, se quiere enseñar cuestiones de *administración* de un gestor de bases de datos (SGBD) real, los alumnos necesitarán privilegios de los que no suelen disponer. Haciendo uso de una máquina virtual sobre la que instalar el SGBD, podrán dedicarse a las tareas de gestión sin problemas. Esto permitiría no sólo crear nuevos usuarios, sino también controlar la organización física de los datos sin restricciones. Naturalmente esta organización repercutirá en el *fichero* que emula el disco lógico, por lo que el rendimiento estará degenerado; pero la posibilidad de hacer este tipo de prácticas puede resultar muy enriquecedora para los alumnos.

Sistemas Operativos es otra asignatura donde el uso de máquinas virtuales es adecuada. La parte teórica no suele verse respaldada por prácticas donde se enseñen cuestiones básicas de administración. De hecho, habitualmente la parte de laboratorio de esta asignatura se desplaza hacia programación de bajo nivel (por ejemplo con POSIX), pero los alumnos nunca llegan, ni siquiera, a crear un nuevo usuario en el sistema.

Con las máquinas virtuales, puede usarse un día de laboratorio para *practicar* la *instalación* de, por ejemplo, alguna distribución de GNU/Linux y explicar el particionado del disco duro, generalmente envuelto en una leyenda de misticismo por el peligro que entraña. Una vez que cada alumno dispone de su *propio* sistema con derechos de administración, las posibilidades se disparan.

Esto último es especialmente claro en las asignaturas posteriores dedicadas al diseño de sistemas operativos. Hoy en día la existencia de diversos sistemas basados en Unix de código abierto permiten guiar la asignatura a través del propio código, por lo que la carga práctica puede ser muy grande. Por desgracia, las restricciones tradicionales en los laboratorios impiden el siguiente paso. Al disponer de máquinas virtuales, sin embargo, se pueden plantear prácticas en las

que los alumnos *modifican y ejecutan* sus propias variantes del núcleo del sistema operativo, o implementaciones de *sus* controladores de dispositivo "de juguete". El uso de una máquina virtual tiene otra ventaja añadida más. Normalmente los errores en la programación de código de tan bajo nivel suele repercutir en un comportamiento errático que desemboca en el bloqueo de la máquina. Sin embargo, al utilizarse software de virtualización el coste por *reiniciar* el sistema no es traumático, y, además, no hay riesgos de pérdida de información por el reinicio reiterado [9].

4. Ejemplo de uso real

Un último uso que, intencionadamente, se ha omitido en la sección anterior es en la asignatura de redes. A parte de la parte teórica de la asignatura, cabría la posibilidad de hacer énfasis en un enfoque *práctico*, donde se describan cuestiones relacionadas con la *instalación y administración* de servicios de redes, como servidores Web o DNS. Como siempre, esto resulta imposible en el modelo habitual de escasez de privilegios por parte de los alumnos. El uso de máquinas virtuales permite solucionar esta carencia.

En este sentido, hemos hecho uso de software de virtualización en la enseñanza del módulo "Redes de Área Local" en el Ciclo Formativo de grado Superior correspondiente al título de Técnico Superior en Administración de Sistemas Informáticos, perteneciente a la Formación Profesional. Si bien dicho título no se enmarca dentro de las enseñanzas universitarias, pensamos que las ideas aquí relatadas podrían utilizarse con éxito también en dicho ámbito, tal y como han descrito las secciones precedentes.

El contenido del módulo (de 290 horas a lo largo de un año académico) debe hacer hincapié en las cuestiones teóricas de las redes de área local, pero también debe proporcionar a los alumnos las habilidades necesarias para su implantación, incluyendo gestión de usuarios y recursos compartidos en Windows y Linux, autenticación remota (Directorio Activo, Kerberos y LDAP), organización de una red de área local en subredes IP, instalación y administración de encaminadores, cortafuegos, etcétera.

Todo esto resulta difícil en el caso de que los alumnos no dispongan de privilegios en la máquina que utilizan. Además, al tener que hacer prácticas relacionadas con la *infraestructura* de la red (más allá de las propias estaciones), las dificultades se incrementan aún más, porque los alumnos deberían poder, por ejemplo, modificar su topología para aprender las repercusiones. El resultado es que ante errores de los alumnos no sólo ordenadores individuales podrían quedar inutilizables temporalmente, sino que la propia red del laboratorio podría resultar afectada.

Una alternativa es crear redes locales paralelas sobre las que se realicen las prácticas. Es el caso del *laboratorio de redes* creado recientemente en la Facultad de Informática de la Universidad Complutense de Madrid. No obstante, las posibilidades no son excesivas y el coste en infraestructura es significativo.

El uso de máquinas virtuales viene, de nuevo, a solucionar el problema, aunque ahora el número de alternativas en la elección del software queda mermado. Las herramientas descritas en la sección 2 permiten la ejecución simultánea de *varias* máquinas virtuales en el mismo anfitrión, pero para poder realizar prácticas de, por ejemplo, encaminamiento IP es, además, necesario *simular una red* entre ellas. En ese caso *no* es suficiente con virtualizar la tarjeta de red, sino que es necesario recrear conexiones "físicas" entre ellas para poder hacer pruebas sin afectar a la red real.

Aunque la última versión de QEmu (Diciembre de 2005) parece que ha hecho un esfuerzo en ese sentido, la única que hoy parece ofrecer posibilidades reales es VMWare. Actualmente es posible *simular redes* entre máquinas virtuales ejecutándose en el mismo anfitrión, hasta el extremo de poder especificar el porcentaje de *errores* en la transmisión. También es posible conectar la tarjeta de red de un huésped a la tarjeta física en modo *puente*, o que el propio VMWare implemente NAT sobre la IP real para proporcionar salida a la red local a las máquinas virtuales.

Con todas estas herramientas, un alumno, en su propio ordenador, puede crear redes arbitrarias de un número reducido de estaciones, configurarlas y administrarlas, sin afectar al funcionamiento de la red real del laboratorio.

Para ilustrar las posibilidades, a continuación se describe un ejemplo de uso llevado a cabo en la impartición del módulo descrito previamente.

Para asentar los conocimientos de la descripción teórica del encaminamiento IP, se decidió que los alumnos implantaran un *router software*

basado en GNU/Linux, en concreto en la distribución Debian. Para ello, se ha utilizado VMWare, donde cada alumno ha aprendido a *instalar* su propia copia de Debian. Cada uno, de hecho, tiene *dos* máquinas virtuales. La primera, que hace las veces del router software, dispone de *dos* tarjetas de red, una de ellas virtualizada, dentro de VMWare, en modo *puente*, por lo que tiene acceso a la red física de forma directa utilizando una IP (estática) diferente a la del anfitrión. La otra tarjeta de red está conectada a una red virtual simulada por VMWare, a la que se encuentra conectada la segunda máquina virtual, para la que el router software se convierte en puerta de enlace.

Con esta organización, los alumnos aprenden a configurar la tabla de rutas del encaminador software, consiguiendo, incluso, que varias de las máquinas virtuales *cliente* que quedan encerradas en las redes simuladas en cada uno de los ordenadores de los alumnos puedan comunicarse entre sí atravesando dos de los routers en Debian. También se realizan prácticas de configuración de cortafuegos (tanto locales en el cliente, como en el servidor para proteger las "redes"), y de instalación de servicios de Internet, como un servidor DNS (bind) o Web (apache).

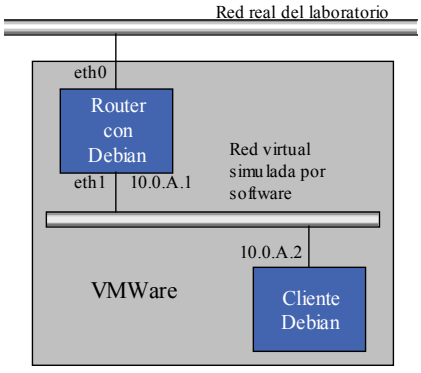

Figura 4. Organización de la red de la práctica

5. Evaluación

La experiencia de uso de VMWare es positiva. El paso más difícil es conseguir que los alumnos *asimilen el concepto* de máquina virtual. Les resultó muy difícil entender que lo que ven dentro de una ventana de VMWare es *exactamente igual* que lo que verían en un ordenador real configurándolo con esas mismas características.

No obstante, superados los primeros titubeos, el resultado es muy positivo. Dado que cada alumno se hace responsable de sus máquinas virtuales, ponen mucho cuidado en mantenerlas al día para poder ir siguiendo el resto de las prácticas. Así todos (de un grupo piloto de tan sólo 10 alumnos) se encargaron de tener a punto sus dos máquinas virtuales y su red privada correctamente funcionando antes de comenzar a realizar las prácticas de cortafuegos.

Además, una vez que los alumnos habían asimilado los diferentes conceptos de forma progresiva sobre sus propias máquinas virtuales, se fue implantando una red real de forma paralela (con un pequeño número ordenadores *físicos*), y los alumnos reaccionaron correctamente a su puesta en marcha gracias a las prácticas realizadas previamente.

Por desgracia, existen, no obstante, potenciales problemas al uso generalizado de este tipo de herramientas. El primero es la memoria física necesaria en el ordenador anfitrión, especialmente si es necesario lanzar varias máquinas virtuales simultáneamente, como en el caso de las prácticas de redes descritas en la sección anterior. Aunque un ordenador con 512 Mb de RAM permite lanzar, con VMWare, un servidor con Windows 2000 Server y un cliente con Windows 2000 Professional, es poco probable que, por ejemplo, pueda soportar un servidor con Windows 2003 Server y un par de clientes con XP. Esto repercute especialmente en la posibilidad de realizar prácticas con los sistemas operativos de Microsoft, dado que las diversas variantes de Linux son más ligeras (al menos en lo que se refiere a servidores) al poderse ahorrar el interfaz gráfico.

Otro problema aparece debido al espacio en disco. Cada máquina virtual requiere un fichero para simular su disco duro, que puede llegar a alcanzar dimensiones de varias centenas de megas. Aunque VMWare permite la *clonación* de máquinas virtuales para intentar minimizar el uso de disco, el resto de herramientas no lo soportan. Dependiendo de la política de uso de disco que se utilice en los laboratorios para los alumnos, este problema podría ser incluso más preocupante que la memoria física. Afortunadamente, en el entorno

en el que se ha realizado la evaluación el espacio físico en disco no resultaba problemático. Además, todas las prácticas se realizaron sobre sistemas huésped basados en GNU/Linux *sin* interfaz gráfica, por lo que el problema de la memoria física no ha surgido en ningún caso.

Por último, las herramientas de código abierto no soportan fácilmente, aún, la simulación de redes internas, a parte de ser algo más complicadas de instalar. VMWare consigue así imponerse como la mejor alternativa, lo que puede llegar a suponer un coste considerable en licencias. Hay una posibilidad para el ahorro, gracias a la existencia del llamado VMWare Player, que permite ejecutar máquinas virtuales creadas por terceros, pero no crear otras máquinas nuevas. Este "visor", distribuido también por VMWare, es gratuito. Para evitar, por tanto, tener que adquirir una licencia de VMWare por cada puesto, podría utilizarse VMWare Player, pero habría que proporcionar máquinas virtuales "vírgenes" a los alumnos de alguna forma. A parte de que esto puede resultar bastante molesto, VMWare Player tiene algunas capacidades de la simulación de redes recortadas, por lo que esta solución podría no ser viable en algunos casos.

6. Trabajo futuro

Aparte de ser usado para las sesiones de prácticas, VMWare también puede utilizarse para la realización de exámenes. En los laboratorios que no estén preparados para esto (porque no hay forma de anular la conexión a Internet, por ejemplo) podrían utilizarse distribuciones Live de GNU/Linux, adaptadas para contener VMWare Player para lanzar el sistema operativo y las aplicaciones necesarias para el examen, restringiendo desde el propio sistema anfitrión la conexión a Internet. Se está analizando esta posibilidad para lanzar simultáneamente varias máquinas virtuales y poder realizar exámenes de redes con contenido práctico *real* de configuración.

7. Conclusiones

El software de virtualización y emulación de la arquitectura PC ha madurado lo suficiente como para poder ser utilizado sin temores. Las restricciones, justificadas, que los alumnos de Informática sufren sobre los ordenadores de los laboratorios impiden a menudo la realización de prácticas pedagógicamente interesantes, por el mero hecho de que requerirían privilegios de administración.

El uso de *máquinas virtuales* se plantea como una buena alternativa para esos casos. El artículo ha enumerado algunas asignaturas donde se podrían utilizar dentro del currículo de Informática, y ha detallado el uso (muy positivo) de este tipo de tecnologías en una clase de uno de los Ciclos de Grado Superior de Informática.

Referencias

[1] R. J. Creasy. The Origin of the VM/370 Time-Sharing System. IBM J. Res. Develop 25(5) pp. 483-490, 1981.
[2] MAME – The Multiple Arcade Machine Emulator. http://www.mame.net/
[3] The Hercules System/370, ESA/390, and z/Architecture Emulator. http://www.conmicro.cx/hercules/
[4] Preboot Execution Environment (PXE) Specification. Disponible en http://www.pix.net/software/pxeboot/archive/pxespec.pdf
[5] bochs: The Open Source IA-32 Emulation Project. http://bochs.sourceforge.net/
[6] QEMU - http://fabrice.bellard.free.fr/qemu/
[7] Microsoft Virtual PC 2004. http://www.microsoft.com/Windows/virtualpc/
[8] VMWare – Virtualization software. http://www.vmware.com/
[9] Jason Nieh and Ozgur Can Leonard, "Examining VMware", Dr. Dobb's Journal, 315, Miller Freeman, San Mateo, CA, August 2000, pp. 70-76