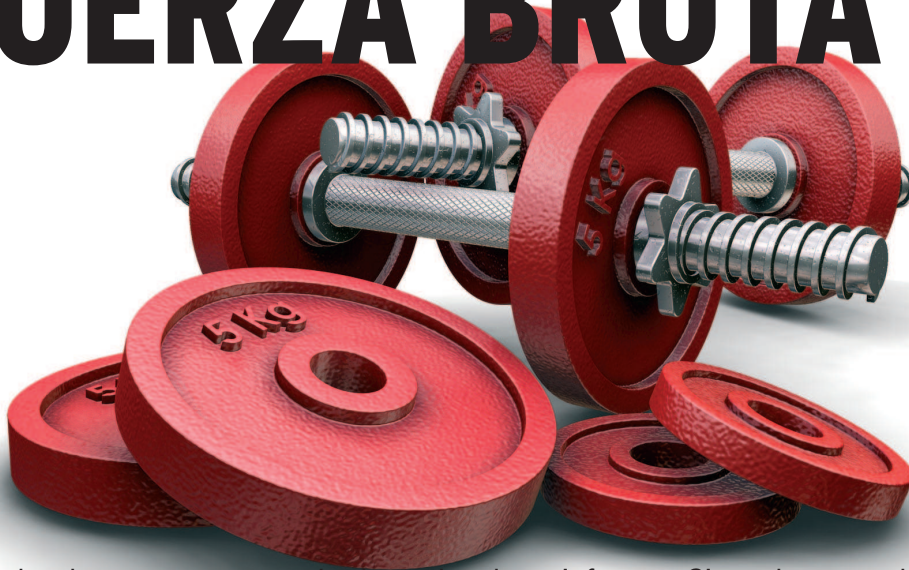


Emulador de mainframe Hercules

FUERZA BRUTA

Kirsty Pargater, Fotolia



Son muchas las empresas que aún dependen de mainframes. Sin embargo, es bastante difícil que podamos echar mano alguno de ellos e instalarles Linux. El emulador Hercules nos plantea una buena alternativa.

POR BERNHARD BABLOK

Los mainframes suelen ser considerados fiables, pero también grandes, complicados, costosos y, actualmente, pasados de moda en el sector de las TI. Por eso nos puede venir bien el editor de mainframes Hercules, que emula la arquitectura de CPU. Incluso si Linux se encuentra en ejecución en un entorno de producción sobre un mainframe, con Hercules podemos tener un útil servicio para las pruebas y el desarrollo.

En este artículo describimos cómo instalar y configurar este emulador y su hardware virtual, para luego centrarnos en la instalación de *zLinux* sobre Hercules. Finalmente, daremos un vistazo rápido a *z/OS* – el sistema operativo nativo para mainframes – y su predecesor disponible de forma libre, *MVS*.

Descarga e Instalación

Hercules 3.04 no soporta las actuales variantes de *zLinux*, motivo por el cual deberemos descargar el código fuente actual de la versión 3.05 [1]. El RPM, disponible también desde la página de inicio, se puede instalar, pero no soporta las conexiones de red y, por tanto, no es apropiado para el propósito de este artículo.

Descomprimos las fuentes en el directorio que queramos (por ejemplo,

/usr/local/src). Cambiamos al directorio */usr/local/src/hercules-3.05* e introducimos la configuración que se muestra en el Listado 1. A continuación no debemos olvidar comprobar *config.log*, prestando especial atención a si disponemos de *Zlib* o no. Si el paquete *zlib-devel* no está instalado en SUSE, aún podemos compilar y usar Hercules, sólo que no podremos utilizar los típicos discos duros virtuales comprimidos con *Zlib*.

Tras completar la fase de configuración seguimos el procedimiento habitual de *make* y *su -c "make install"*. Ya está instalado Hercules, pero hay que hacer algo más. La documentación en formato *HTML* se encuentra en */usr/local/share/hercules*, con enlaces a otros recursos en Internet. Si tenemos algún problema, siempre podemos consultar la lista de correo.

Es importante configurar el sistema anfitrión correctamente siempre que planeemos usar redes TCP/IP en el sistema huésped. Primero tenemos que modificar los privilegios del script */usr/local/bin/hercifc*, que configura el dispositivo de red Tun/Tap al inicio de Hercules. Igual que con el acceso directo a cualquier dispositivo, necesitamos privilegios de *root*. La opción de configuración *—enable-setuid-hercifc* (ver Listado 1, línea 9) activa el bit *SUID*, pero como el

script de instalación coloca *root* como grupo del archivo, no se puede ejecutar el script como usuario sin privilegios. Las si líneas que siguen establecen los permisos adecuados:

```
# chgrp users ➤
/usr/local/bin/hercifc
# chmod 4750 ➤
/usr/local/bin/hercifc
```

Las siguientes secciones sólo son aplicables a sistemas *openSUSE*; algunos detalles varían entre los distintos sistemas operativos. Para la conexión de red, Hercules utiliza el dispositivo Tun/Tap; en un sistema normal, el dispositivo pertenece a *root:root* y tiene como permisos *0600*. Una vez más, esto impide el acceso a usuarios no privilegiados. Cambiar los permisos manualmente no nos servirá de mucho, ya que al reiniciar se restaurará el estado original de los mismos.

Los sistemas actuales utilizan *Udev* para la gestión de los dispositivos, motivo por el cual tenemos que modificar la correspondiente regla. En un sistema *openSUSE*, es el archivo */etc/udev/rules.d/50-udev-default* quien lo controla, por lo que debemos añadir:

```
KERNEL=="tun", GROUP=="users", ➤
NAME=="net/%k",MODE=="0660"
```



Figura 1: Un salto atrás en el tiempo: una impresora IBM 1403 por líneas.

Para dar a `/dev/net/tun` los permisos adecuados, introducimos `rm /dev/net/tun y modprobe tun`. Si el dispositivo sigue teniendo mal los permisos después de reiniciar, no hay que preocuparse. En el momento de arrancar el sistema, `/etc/init.d/boot.udev` copia una serie de dispositivos de `/lib/udev/devices` a `/dev`, por lo que hay que eliminar el dispositivo `Tun` de este directorio o modificar sus permisos:

```
chgrp users z
/lib/udev/devices/net/tun
chmod 660 z
/lib/udev/devices/net/tun
```

Preparación de la Red

El dispositivo `Tun/Tap` soporta una conexión punto-a-punto entre anfitrión y huésped, lo que otras soluciones de virtualización llaman *host-only networking*. Si queremos garantizar el acceso a la red local del sistema huésped – o incluso a Internet – tendremos que hacer también

Listado 1: Compilación de Hercules: Argumentos de Configuración

```
01 #!/bin/bash
02 PREFIX=/usr/local
03
04 ./configure --prefix=$PREFIX \
05             --enable-cckd-bzip2
06             \
07             --enable-optimization=yes \
08             --disable-external-gui \
09             --enable-setuid-hercifc
```

NAT. Para ello activamos primero la redirección de IP permanentemente en `/etc/sysconfig/sysctl` (`IP_FORWARD=yes`) o manualmente:

```
echo "1" > z
/proc/sys/net/ipv4/ip_forward
```

Luego definimos dos nuevas reglas en `iptables`:

```
iptables -t nat -A
POSTROUTING z
-o eth0 -j MASQUERADE
iptables -A FORWARD -i z
tun0 -j ACCEPT
```

Ya está preparado el sistema anfitrión. Hemos llegado a la configuración del hardware virtual.

Hardware Virtual

Hercules soporta una buena cantidad de componentes de hardware, pero para *zLinux* sólo vamos a necesitar unos discos duros y una conexión de red. Debido a que se trata de un sistema de servidor sofisticado, el soporte para vídeo, audio y USB es irrelevante. Aunque Hercules soporta la emulación de impresoras, no hay controladores para Linux para la antigua impresora 1403 orientada a líneas (Figura 1). Tampoco es un inconveniente demasiado importante; siempre podemos imprimir en red al servidor `CUPS` del anfitrión o a cualquier otra impresora de la red.

El hardware virtual se define en un archivo de configuración (Listado 2), que se pasa al emulador mediante el parámetro `-f` en el arranque de éste. Hercules no impone restricciones en cuanto al nombre o la estructura del directorio. Dependerá de nuestras preferencias personales el que pongamos todos los archivos en un solo directorio o los organicemos en subdirectorios por tipo de archivo.

Los discos duros virtuales son archivos normales, como con cualquier otro emulador. Hablando en lenguaje IBM, nos referimos a los discos duros como `DASDs` (*direct-access storage devices*). Para crear discos duros en nuestra instalación de *zLinux*, introducimos:

```
dasdinit -z -lfs z
dasd/sys1.dasd 3390-3 SYS1
dasdinit -z -lfs z
dasd/sys2.dasd 3390-3 SYS2
```

La opción `-z` configura la compresión con `Zlib`. Alternativamente, podemos usar `-bz2` para elegir una compresión `Bzip2`; nótese que este método es más lento, con lo que tendríamos una merma en el rendimiento sin ahorrar mucho más espacio en disco. La opción `-lfs` indica soporte para archivos de gran tamaño; esto es, `dasdinit` no partirá el disco duro en trozos de 2GB. Dependiendo del tamaño de nuestro `DASD`, podemos omitir esta opción y guardar los archivos en DVDs.

Los nombres de archivo (`dasd/sys?.dasd`) son arbitrarios. El tipo de disco duro 3390-3 define el tamaño de éste, que en nuestro ejemplo es de unos 2.7GB. En la documentación de Hercules tenemos un listado de los tipos disponibles y sus respectivos tamaños [2].

El último parámetro pasado a `dasdinit`, el nombre `HERCULM` (la etiqueta del disco), en realidad no hace falta en Linux, aunque la instalación escribe esta etiqueta de todos modos. Como último paso, el administrador tiene que añadir los discos al archivo de configuración de Hercules (Listado 2, líneas 29-30), asignándose ahí las direcciones de los dispositivos para que Linux referencie los discos.

Interfaz de Red

Hercules emula varios tipos de conexiones de red. El tipo más fácil de configurar

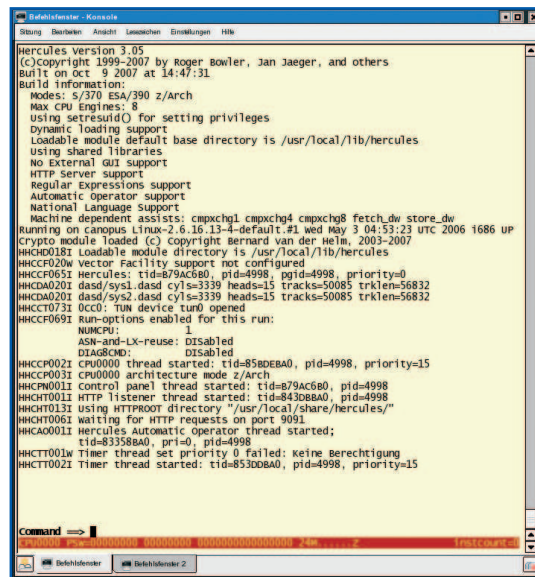


Figura 2: La consola de Hercules mostrando al inicio las funcionalidades soportadas.

Listado 2: Configuración de Hercules para CentOS 4.5

```

01 #
02 # Archivo de configuración para Hercules & CentOS
   4.5 s390
03 #
04
05 MODPATH
   ${MODPATH_HERCULES:=/usr/local/lib/hercules}
06
07 CPUSERIAL 002623
08 CPUMODEL  2096
09 MAINSIZE  780
10 XPNDSIZE  0          # Almacenamiento expandido
   en megabytes
11 CNSLPORT  3270      # puerto TCP al que se
   conectan las consolas
12 HTTPROOT
   ${HTTPROOT_HERCULES:=/usr/local/share/hercules/}
13 HTTPPORT  9091      # servidor HTTP
14 NUMCPU    1          # Número de CPUs
15 NUMVEC    1          # Instalaciones de vector
   emuladas
16 SYSEPOCH  1900
17 TZOFFSET  +0000
18 OSTAILOR  LINUX     # confección del SO
19 PANRATE   SLOW      # Frecuencia de
   actualización del panel
20 ARCHMODE  ESAME     # Mo de arquitectura
   S/370, ESA/390 o ESAME
21 PGMPRDOS  RESTRICTED
22
23
24 # Listado de Dispositivos
25 # --  —  _____
26
27 # DASD
28
29 OA01 3390      dasd/sys1.dasd
30 OA02 3390      dasd/sys2.dasd
31
32 # Canal a canal: CTCI   ip-linux-huésped
   ip-linux-anfitrión
33
34 0cc0.2 3088 CTCI 192.168.2.2 192.168.2.1

```

es la conexión *canal-a-canal* (CTC). Para configurar dos dispositivos CTC de tipo 3088 en las direcciones *0cc0* y *0cc1* sólo tenemos que añadir una línea al archivo de configuración (Listado 2, línea 34) y asignar la dirección *192.168.2.2* al huésped y *192.168.2.1* al anfitrión. La documentación en *HTML* muestra la sintaxis

para direcciones de dispositivos en particular.

Para iniciar Hercules introducimos el comando:

```

hercules -f conf/ >
centos45.conf >
> hercules.log

```

La Figura 2 muestra la salida inicial en la consola de Hercules. Aquí, el mensaje crítico es *HHCC073I*, que nos indica que el dispositivo de red *tun0* está abierto. Desgraciadamente, el mensaje no es del todo fiable, ya que depende de los permisos de *hercifc*; para poder estar seguros, hemos de comprobar la salida de los comandos *route -n* e *ifconfig* (Listado 3).

Tanto la tabla de enrutamientos (línea 4ff) como la configuración de red (líneas 29-35) deben tener entradas para el dispositivo *tun0*.

Listado 3: Verificando la Configuración de la Red

```

01 [root@sirius:~] # route -n
02 Kernel IP routing table
03 Destination      Gateway            Genmask           Flags Metric Ref    Use
   Iface
04 192.168.2.2      0.0.0.0           255.255.255.255  UH        0      0
   0 tun0
05 ...
06 127.0.0.0        0.0.0.0           255.0.0.0         U          0      0
   lo
07 [root@sirius:~] # ifconfig
08 eth0             ...
09
10 lo               ...
11
12 tun0             Link encap:UNSPEC HWaddr
13 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
14                  inet addr:192.168.2.1  P-t-P:192.168.2.2
   Mask:255.255.255.255
15                  UP POINTOPOINT RUNNING MTU:1500 Metric:1
16                  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
17                  TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
18                  collisions:0 txqueuelen:500
19                  RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

```

Uso de Hercules

La consola de Hercules sirve para controlar el sistema. Hercules soporta un buen número de comandos, de los cuales el más importante es *ipl* (*initial program load*), que inicia el proceso de arranque. El comando *maxrates* también es útil; muestra el rendimiento en *MIPS* (millones de instrucciones por segundo).

Hercules soporta otras consolas. Al pulsar la tecla *Esc* alternamos entre la consola orientada a líneas de la Figura 1 y la consola semi-gráfica mostrada en la Figura 3. Esta vista nos informa del rendimiento actual y nos muestra en pantalla los valores de los registros.

Para ejecutar Hercules de forma remota podemos usar también una consola por *http* (Figura 4), que el administrador puede definir en el archivo de

configuración (Listado 2, líneas 12-13). Además, podemos especificar nombre de usuario – y autenticación basada en contraseña, pero no cambia el hecho de que la conexión por *http* es insegura.

Instalación de zLinux

Los mainframes soportan dos arquitecturas de sistema distintas: zLinux 31 bits (*s390*) y zLinux 64 bits (*s390x*). Sólo el direccionamiento es de 31 bits en la variante de 31 bits; los datos en sí utilizan 32 bits.

Si estamos buscando un *zLinux* libre de 64 bits, podemos optar por ThinkBlue/64 Linux [3] o el nuevo CentOS 4.5 [4], que es la distribución usada para este artículo.

Debian ofrece una buena alternativa; la arquitectura de 31 bits se mantiene a lo largo de sus distintas publicaciones. Aquí no explicaremos las diferencias entre las instalaciones de Debian y CentOS.

Instalación

La instalación de *CentOS* en el mainframe se puede hacer a través de cable. Debido a que los mainframes y Hercules no soportan dispositivos de CD, es necesario exportar el DVD, montado a través de *NFS*, o copiar el contenido a un directorio exportado. Al introducir *ipl /cdrom/generic.ins* se arranca el sistema desde la consola de Hercules.

A continuación deberíamos ver los mensajes habituales del arranque de Linux aparecer en pantalla. Lo siguiente es iniciar el programa de configuración de la red, que nos pide la información necesaria para la configuración de una conexión. El Listado 4 muestra los mensajes y las respuestas. Nótese que la consola de Hercules pasa comandos que comienzan

Listado 4: Instalación de zLinux: Primera Fase

```
01 Which kind of network device do you intend to use:
02 .ctc
03 Enter the bus ID and the device number of your CCW devices.
04 CTC/ESCON and LCS need two subchannels:
05 (e.g. "0.0.0600,0.0.0601" will configure the CTC or ESCON interface
06 with the subchannels 0x600 and 0x601)
07 QETH needs three subchannels p.e. 0.0.0300,0.0.0301,0.0.0302
08 .0.0.0cc0,0.0.0cc1
09 Enter the FQDN of your new Linux guest (e.g. s390.redhat.com):
10 .emu-guest.bablokb-local.de
11 Enter the IP address of your new Linux guest:
12 .192.168.2.2
13 Enter the network address of the new Linux guest:
14 .192.168.2.0
15 Enter the IP of your CTC / ESCON / IUCV point-to-point partner:
16 .192.168.2.1
17 Select which protocol should be used for the CTC interface
18 0 for compatibility with p.e. VM TCP service machine (default)
19 1 for enhanced package checking for Linux peers
20 3 for compatibility with OS/390 or z/OS peers
21 .0
22 Enter your DNS server(s), separated by colons (:):
23 .10.173.112.250
24 Enter your DNS search domain(s) (if any), separated by colons (:):
25 .
26 Enter DASD range (e.g. 200-203 or 200,201,202,203)
27 Press <Enter> for autoprobing (not recommended):
28 .0a01-0a02
29
30 Starting telnetd and sshd to allow login over the network.
31
32 Connect now to 192.168.2.2 to start the installation.
```

por un punto a través del sistema en ejecución.

Tras completar esta fase aparece el mensaje mostrado en la línea 32. Ya podemos establecer una conexión por SSH con la dirección del huésped, que nos lleva a la herramienta estándar de instalación de *CentOS*: *Anaconda*. Después de cambiar algunas configuraciones y de montar

Con un mainframe real el proceso es bien diferente, ya que podemos optar entre usar HMC (*hardware management console*) y cintas de arranque, o preparar un volumen para el *zLinux* del lado del *z/OS*.

Tras la Instalación

El comando *ipl* arrancará un sistema instalado en Hercules. Como parámetro, Hercules espera que se le pase la dirección del volumen de arranque (*ipl 0a01*, en nuestro ejemplo). Después de *IPL* aparece un menú al estilo de grub, aunque el sistema arrancará automáticamente pasados 15 segundos. La consola muestra los típicos mensajes de inicio de un sistema Linux (*Red Hat*) hasta llegar al prompt del login. No es muy práctico trabajar directamente en la consola, porque hay que escribir un punto antes de cada comando y porque algunos programas no son ejecutables en consola. Lo más lógico es usar una conexión por *SSH* desde otra máquina.

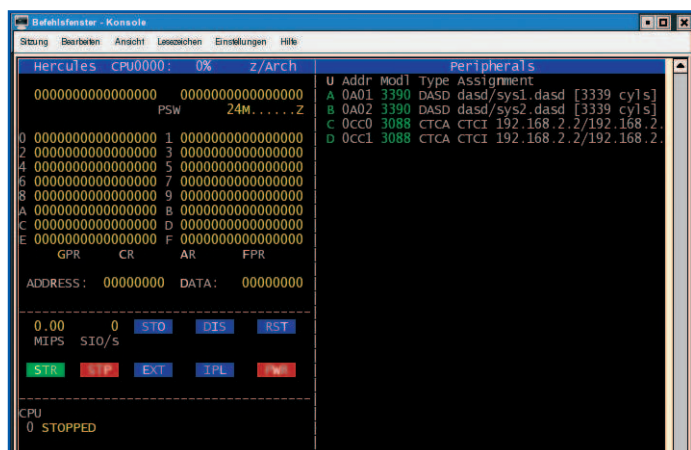


Figura 3: Hercules en modo semi-gráfico con su consola rica en colores.

Adición Dinámica de Discos

En principio, *zLinux* no se diferencia del Linux de cualquier otra plataforma, excepto por la forma en que gestiona el hardware. El siguiente ejemplo monta otro disco en */home* en el sistema, pero antes hay que generar el nuevo disco (el archivo *dasd/home.dasd*). Hercules emula un sistema de alta disponibilidad; el administrador no tiene que apagar *zLinux*, sino que se puede usar la consola de Hercules para introducir el nuevo dispositivo:

```
attach 0a03 ↵
3390 dasd/ ↵
home.dasd
```

En la página de *zLinux*, al teclear *lsdasd -s -a* se nos muestra un listado con los dispositivos DASD disponibles; podemos ver el nuevo dispositivo, aunque desconectado. Mientras que el operador de un mainframe introduciría el comando *vary online*, root haría lo siguiente:

```
echo 1 > /sys/bus/ccw/ ↵
drivers/dasd-eckd/ ↵
0.0.0a03/online
```

Con *lsdasd* y echando un ojo en el directorio */dev*, vemos que ahora existe *dasdc*.

Seguidamente tenemos que hacer un formato a bajo nivel del dispositivo y particionarlo: *dasdfmt* se encarga de la primera de estas tareas, mientras que *fdasd* lo hace de la segunda:

```
dasdfmt -p -v -l HOME01 -b ↵
4096 -d cd1 -f /dev/dasdc
fdasd -a /dev/dasdc
mke2fs -j /dev/dasdc
mount /dev/dasdc /home
```

Si estamos configurando múltiples particiones, tenemos que llamar a *fdasd* con el parámetro *-a*, lo que nos lleva a un menú de particionamiento al estilo de *fdisk*.

En el ejemplo usaremos el volumen completo como una partición; de todos modos, el nuevo dispositivo se pierde al reiniciar. Para evitarlo tenemos que añadir una línea para el dispositivo en el archivo de configuración de Hercules y añadir una entrada a */etc/fstab* para su montaje.

El módulo del kernel *dasd_mod* espera que se le pasen los dispositivos disponibles como opción de inicio; lo hacemos cambiando */etc/modprobe.conf*:

```
options dasd_mod ↵
dasd=0a01-0a03
```

Para iniciar el sistema necesitamos un nuevo *Initrd*. Para no correr riesgos innecesarios, copiamos el *Initrd* existente y añadimos una entrada en el menú de arranque, en */etc/zipl.conf*, para poder arrancar la versión anterior en caso de error:

```
cd /boot
mv initrd-2.6.9-55.EL.img ↵
initrd-2.6.9-55. ↵
EL.img.orig
mkinitrd -v initrd-2.6.9-55.EL. ↵
img 2.6.9-55.EL
vi /etc/zipl.conf
# -> Modificar la carga del ↵
initrd original
zipl -V
```

Este procedimiento es similar al del cargador de arranque tradicional de Linux, *lilo*, con el que tenemos que ejecutar el programa después de modificar el archivo de configuración.

Sin Snapshots

Hercules no usa snapshots como hacen otras soluciones de virtualización, pero implementa archivos *shadow*, que se pueden usar de forma similar. Para usar un archivo *shadow* hay que cambiar lige-

El Principio del Mainframe

Aunque Linux aún se basa en varios programadores para asegurarse una gestión de los recursos óptima, el mundo del mainframe ha seguido su propio camino. En un mainframe son los usuarios quienes especifican los recursos que necesitan, lo que permite al sistema decidir si el programa debería ejecutarse ahora, después, o no hacerlo. El Listado 5, un programa de copia típico, parece bastante más complicado que el comando *cp*, pero dicha complejidad tiene tantos inconvenientes como ventajas.

Los trabajos de un mainframe no son mucho más que una pila de tarjetas perforadas enlazadas a un archivo. El trabajo aquí impreso comienza con una tarjeta que especifica la memoria, el tiempo de ejecución y el tipo de trabajo, lo que facilita su priorización. Luego hay que copiar el programa, IEBGENER, y sus parámetros. Los programas del host usan nombres de archivo lógicos para acceder a los archivos; para IEBGENER, estos nombres – definidos por tarjetas DD (*data definition*) – son *SYSUT1* y *SYSUT2*.

SYSUT1 (línea 5), el archivo de entrada, se designa como “OLD”, lo que significa que lo necesitamos en exclusiva. Si hay otro trabajo procesando este mismo archivo, el nuestro tendrá que esperar. Una alternativa sería *DISP=SHR*, que permite acceso (de lectura) compartido (y podría ser una alternativa más realista para un programa de copia).

Nótese la especificación del archivo de destino en las líneas 6 a la 9. (*NEW,CATLG*) estipula que el archivo no debe existir y que debería ser catalogado después de su correcto procesamiento. Si el archivo existe, el trabajo no se ejecutará. Los otros detalles especifican los requerimientos de espacio (5 cilindros primarios, adicionalmente hasta 15 veces 2 cilindros) y el formato del archivo físico. En Linux, un programa malicioso puede llenar completamente el sistema operativo; en un mainframe, el programa se detiene si sobrepasa la asignación de espacio.

La optimización conlleva un esfuerzo adicional. En el pasado, cuando los

mainframes mes eran mucho más pequeños, este tipo de salvaguarda era más importante, mientras que los mainframes de hoy día incluyen opciones que simplifican el proceso. El objetivo de maximización del uso no ha cambiado desde entonces. Los valores de utilización del 90 por ciento en operaciones 24x7 siguen siendo posibles, algo financieramente muy lógico, por otro lado.

El protocolo que utilizan las terminales de mainframe para comunicarse con el anfitrión es otro ejemplo del excelente y máximo aprovechamiento de recursos que utilizan estas máquinas. Las terminales físicas ya no existen; programas como el *x3270* emulan el protocolo. En contraste a los controladores de teclado de Unix, que procesan cada pulsación de teclado, el usuario de un terminal *3270* rellena todos los campos de la pantalla para luego transmitirlo todo de una vez. De este modo se reduce el número de pasos en el procesamiento, con un coste añadido que es el no poder utilizar programas como *vi* o *emacs*.

ramente las líneas de la configuración del disco duro (Listado 2, línea 29ff.):

```
0A01 3390 dasd/sys1.dasd ro 2
sf=shadow/sys1_*.dasd
0A02 3390 dasd/sys2.dasd ro 2
sf=shadow/sys2_*.dasd
```

Después de iniciar, Hercules escribe automáticamente los archivos *shadow/sys?_1.dasd*. El comando *sf+ **, en la consola de Hercules, crea un juego nuevo de archivos *shadow*, hasta un máximo de ocho archivos. *sf- *merge* o *sf- *nomerge* acepta o descarta los últimos cambios producidos. Aceptarlos implica que *sf-merge* escriba los cambios desde, por ejemplo, *sys1_2.dasd* a *sys1_1.dasd*. Para escribir los cambios a los archivos originales hemos de especificar *sf- *force* (u omitir la opción *ro* de sólo lectura).

Los comandos *suspend* y *resume* guardan y restauran el estado completo del sistema; de todos modos, este mecanismo no es del todo fiable – *zLinux* no funcionará después de usarlo.

z/OS y MVS

Desde la perspectiva de Linux, el mainframe tan sólo es una arquitectura más, por lo que parece lógico echar un vistazo a los sistemas operativos que se ejecutan normalmente en los mainframes. Entre todos los existentes, el sistema por excelencia es *z/OS 1.x* (La versión 1.9 se publicó en Septiembre de 2007).

Por desgracia, *z/OS* no es libre. Aunque hay un paquete *ADCD* para desarrolladores y también un paquete demo de *z/OS*, sin embargo necesitamos al menos contratar una máquina IBM para poder usarlo legalmente. Con ello, el abuelo del *z/OS* de hoy día, *MVS*, es el único que queda con el que podemos experimentar en casa.

MVS (Multiple Virtual Storage) fue desarrollado a partir de los antiguos sistemas operativos de IBM; su antecesor directo es el *OS/360 MVT (Multitarea, con un número variable de Tareas)*. Luego, *MVS* se convirtió en el sistema operativo de la siguiente generación de hardware, la máquina *System/370*, que soportaba memoria virtual desde 1972.

Es curioso que los modelos antiguos de la *S/360-67* ya soportaban memoria virtual, pero por ese tiempo el proceso por lotes se consideraba mucho más importante, eliminándose de las series *S/370* el hardware que soportaba la memoria virtual.

MVS salió al mercado en 1974 y fue líder a través de *MVS/XA*, *MVS/ESA* y *OS/390* a *z/OS*. Ahora *MVS* está disponible libremente, ya que los sistemas operativos de aquel tiempo se distribuían abiertamente.

Hay un sistema *MVS* disponible para Hercules: *MVS Turnkey System* [7][8]. El único software adicional necesario es un emulador de terminales *3270*; los usuarios de Linux pueden instalar el paquete *x3270* de cada distribución.

Quien crea que puede instalar un sistema mainframe rápidamente y empezar a usarlo se equivoca. Aunque consigamos arrancar el sistema usando alguna receta [8], no nos servirá demasiado si no tenemos algo de experiencia previa con mainframes. Todo aquel que esté interesado de verdad en sumergirse en este mundo, trabajando con la documentación disponible – incluyendo la

documentación del actual *z/OS* – de seguro disfrutará de la experiencia de tener su propio mainframe.

Conclusiones

Hercules ofrece un emulador estable para la arquitectura de mainframe. Su ejecución en un sistema potente de gama media dará un rendimiento suficiente como para reemplazar un *LPAR* de *zLinux* de pruebas. Quien esté pensando en portar su software a *zLinux* puede utilizar tanto Hercules como *LCDS (Community Development System for Linux)*, un servicio proporcionado por IBM [9]. ■

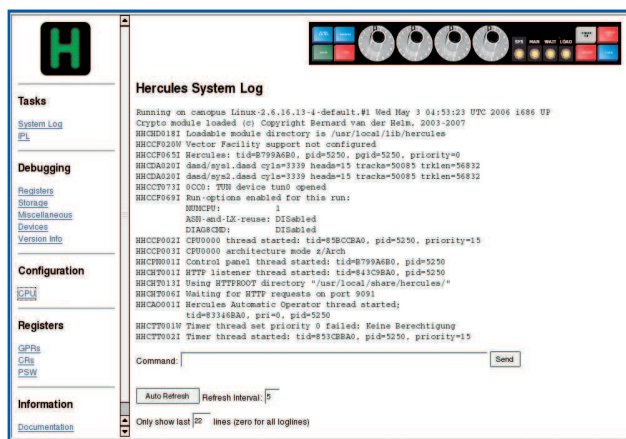


Figura 4: La consola web de Hercules.

RECURSOS

- [1] Hercules: <http://www.hercules-390.org>
- [2] Tabla de conversión para tamaños de DASD: http://www.sdisw.com/vm/dasd_capacity.html
- [3] ThinkBlue/64 Linux para zSeries: <http://linux.zseries.org>
- [4] CentOS: <http://www.centos.org>
- [5] Maclsaac, Michael, Ronald Anuss, Wolfgang Engel, et al. *Linux for IBM eServer zSeries and S/390: Distributions*. Redbooks, 2001: <http://www.redbooks.ibm.com/redbooks/pdfs/sg246264.pdf>
- [6] Portal de Linux Redbook: <http://www.redbooks.ibm.com/Redbooks.nsf/portals/Linux>
- [7] MVS 3.8j Turnkey System: <http://www.ibiblio.org/jmaynard/turnkey-mvs-3.zip>
- [8] Información sobre MVS Turnkey System: <http://www.bsp-gmbh.com/turnkey/index.html>
- [9] LCDS (Community Development System for Linux): <http://www-03.ibm.com/systems/z/os/linux/lclds>

Listado 5: Copiando un Archivo con IEBGENER

```
01 //MYJOB JOB (COPY), 'COPY', CLASS=A,MSGCLASS=A,REGION=256K,
02 // TIME=5,MSGLEVEL=(1,1),NOTIFY=HERC04
03 //COPY EXEC PGM=IEBGENER
04 //SYSPRINT DD SYSOUT=*
05 //SYSUT1 DD DISP=OLD,DSN=HERC04.DATEI2
06 //SYSUT2 DD DSN=HERC04.FILE2.BAK,DISP=(NEW,CATLG),
07 // UNIT=SYSDA,VOL=SER=PUB002,
08 // SPACE=(CYL,(5,2,0)),
09 // DCB=(RECFM=FB,LRECL=80,BLKSIZE=4096,DSORG=PS)
10 //SYSIN DD DUMMY
```