

Guía para la Instalación de MINIX

(Sobre UNIX)

Laboratorio de Sistemas Operativos

Guía para la Instalación de MINIX	1
Introducción	1
Instalación	1
Compilación del BOCHS	1
Generación de discos	1
Generación de archivos configuración para Bochs	2
Instalación de MINIX sobre bochs	2
Backups del Sistema	3
Intercambios de archivos entre Unix y Minix	3
ANEXO: Fuente del comando pad	4

Introducción

Para poder ejecutar el Minix sobre un sistema UNIX, en particular LINUX, se debe usar un simulador de la arquitectura Intel x86 y dentro de dicho emulador ejecutar el minix.

El simulador emula una computadora completa y permite que sobre la misma se ejecute cualquier sistema operativo, como ser Linux, Windows, DOS, Minix, etc.

En el CD distribución de MINIX 2.0 viene un simulador llamado BOCHS. Una versión generada en 1996, la cual luego de compilar permite ejecutar el minix sin problema alguna. En la última versión del BOCHS, julio de 2002 (bochs-1.4.1), al bootear el Minix genera un error sobre la imposibilidad de acceder al root fs.

En el archivo linminix.zip se encuentra la versión de bochs-1.3.0, que fue probada y permite ejecutar sobre ella el minix. Se recomienda su uso, dado que permite también simular arquitecturas x86 de mas de un procesador.

Instalación

En el archivo linminix.zip, se encuentran los fuentes de bochs comprimidos en el archivo **bochs-1.3.tar.gz**, un archivo de configuración para BOCHS (**.bochsrc**) y una imagen del disquete de inicio de Minix 1.44M.

Una vez descomprimido el archivo linminix.zip y el archivo bochs-1.3.tar.gz, se procederá a la compilación del simulador. Cabe señalar que las versiones precompiladas del bochs no fueron generadas con las opciones necesarias para ejecutar el MINIX adentro.

Compilación del BOCHS

Para generar el simulador, viene en la distribución un archivo llamado configure. A dicho comando se lo debe ejecutar con las siguientes opciones:

```
/home/usuario/bochs$ ./configure --enable-bochs-bios-hooks --enable-dma-floppy-io
```

El comando anterior genera los archivos Makefiles necesarios y el archivo config.h donde se encuentran las opciones para el simulador. Para obtener los archivos ejecutables del simulador basta con ejecutar el siguiente comando:

```
/home/usuario/bochs$ make bochs
```

Esto solo generará los archivos del simulador dado que la distribución de bochs trae también un mini linux para hacer pruebas sobre el simulador.

Generación de discos

El simulador permite definir el hardware que tendrá la máquina virtual simulada, mediante el uso de un archivo de configuración. Dentro de dicho archivos se indican entre otras cosas el archivo que simulará al disco rígido. Cabe señalar que dicho archivo deberá contener el tamaño necesario para contener la información completa según la definición de los parámetros sectores y pistas. A continuación se indica el comando en unix para generar archivos llenos de zeros y los tamaños correspondientes a algunos parámetros:

Para generar un disco de 112 Mb

```
$ dd if=/dev/zero of=112M bs=512 count=229500
```

Para generar un disco rígido de otro tamaño se debe cambiar el parámetro **count** según la siguiente tabla.

Tabla con los tamaños y parámetros para los discos rígidos:

Tamaño	Cilindros	Cabezas	Sectores	Total
10MB	306	4	17	20808
20MB	615	4	17	41820
30MB	615	6	17	62730
46MB	940	6	17	95880
62MB	940	8	17	127840
112MB	900	15	17	229500
483MB	1024	15	63	967680

Generación de archivos configuración para Bochs

En el archivo `linminix.zip`, existe un archivo con la configuración básica para el uso del `minix` sobre `bochs`.

1	<code>megs: 16</code>
2	<code>diskc: file=./112M, cyl=900, heads=15, spt=17</code>
3	<code>floppya: 1_44=./1.44M</code>
4	<code>log: ./bochs.out</code>
5	<code>vga_update_interval: 250000</code>
6	<code>keyboard_serial_delay: 200</code>
7	<code>vgaromimage: /home/usuario/bochs/bios/VGABIOS-elpin-2.40</code>
8	<code>romimage: file= /home/usuario/bochs/bios/BIOS-bochs-latest, address=0xf0000</code>
9	<code>floppy_command_delay: 150000</code>

Los comandos de configuración del `bochs` indicados anteriormente son buscados por el simulador en varios lugares, siendo uno de ellos la ubicación desde donde se invoca a `bochs`. Para hacer uso de la disquetera directamente sin tener que generar una imagen del disquete en el archivo `1.44M` indicado en la configuración se puede modificar la línea de `floppya`: a como sigue:

```
floppya: 1_44=/dev/fd0
```

Esto hará que se trabaje sobre la disquetera directamente. En caso de querer usar imágenes de disquetes se puede usar el siguiente comando:

```
$ dd if=/dev/fd0 ibs=512 of=1.44M
```

Instalación de MINIX sobre bochs

Una vez instalado el simulador, se lo debe invocar desde el directorio donde se generaron las imágenes del disco rígido y el disquete:

```
$ /home/usuario/bochs/bochs
```

Aparecerá un menú donde se pide que se indique el archivo de configuración [1]. Siendo este el archivo `.bochsrc` del subdirectorío actual. Es posible modificar los seteos de dicho archivo y por último se lanza al simulador con la opción [4].

La instalación de `Minix`, es igual a la instalación con partición de disco rígido indicada en el documento **Notas sobre la instalacion de MINIX particionando el Disco Rígido** de la página de la materia.

Una vez instalado el `minix`, será necesario bootear desde el disco rígido y eso se logra utilizando la opción `boot:c` del archivo de configuración `.bochsrc`. También es posible utilizar la opción de la línea de comandos.

Backups del Sistema

Siguiendo el mismo pensamiento, se puede realizar backups completos, haciendo valer que todo el disco rígido usado por minix, es un archivo de Unix, el cual puede ser comprimido usando **gzip**. Pudiendo dejar los backups en el disco de Unix o bajarlo a disquetes.

Cuando se comprimen los archivos de simulación de disco, se debe tener en cuenta que los sectores en dicha simulación contienen datos aún cuando los archivos hayan sido borrados por el minix y por ello, el archivo comprimido va a ser mas grande. Para lograr reducir este efecto, se puede ejecutar el siguiente script dentro de minix:

```
cd /usr/tmp
echo >junk
while cat junk >>junk; do ;; done
sync
rm junk
```

Y luego realizar la compresión del archivo desde el shell de Unix. El script anterior llena de LF a todos los bloques que se encuentran eliminados en el filesystem de minix, repercutiendo esto en el factor de compresión que se obtiene al usar el **gzip**.

Queda también la posibilidad de realizar una copia de los archivos fuentes a disquetes como se detallo en el documento de *Guía para la compilación de MINIX*.

Intercambios de archivos entre Unix y Minix

Para llevar datos de un directorio (DIRE) de Unix al Minix se puede utilizar los siguientes comandos:

```
Unix$ tar cf - DIRE > DIRE.tar
Unix$ compress DIRE.tar
Unix$ pad 1474560 < DIRE.tar.Z > 1.44M-b
```

Luego editar el archivo .bochsrc para poner el archivo 1.44M-b como uno de los floppys (floppyb:) y luego cargar a bochs y minix. Desde el prompt de Minix ejecutar el siguiente comando:

```
Minix% vol -r /dev/fd1 | uncompress | tar xf -
```

Y listo. Ya está disponible en Minix el subdirectorio DIRE. Mientras que la **vuelta** es similar y se obtiene usando los siguientes comandos:

```
Minix% tar cf - DIRE > DIRE.tar
Minix% compress DIRE.tar
Minix% vol -w /dev/fd1 < DIRE.tar.Z
```

Luego en Unix se obtiene la información con el siguiente comando:

```
Unix$ uncompress -c 1.44M-b | tar xf -
```

NOTA:

Del procedimiento anterior el comando **pad**, no es un comando estándar de Unix cuyo código se encuentra en el anexo.

ANEXO: Fuente del comando pad

```
---- pad.c ----
#include<stdio.h>
#include<stdlib.h>

void print_usage (void) {
    fprintf(stderr,"Usage:\n");
    fprintf(stderr," pad SIZE < input.file > output.file\n");
    fprintf(stderr,"copies stdin to stdout then outputs enough 0's to make the total size\n");
    fprintf(stderr,"of stdout equal to SIZE (in bytes)\n");
    fprintf(stderr,"\nAuthor: Bradley C. Kuszmaul, bradley@cs.yale.edu");
    fprintf(stderr,"Copyright 1997 Yale University.\n");
    fprintf(stderr,"There is no warranty for this free software.\n");
    fprintf(stderr,"Distributed under the terms of the Gnu Public\n");
    fprintf(stderr," License, available in the usual places on the web.\n");
    fprintf(stderr," And there ain't no insanity clause.\n");
}

main (int argc, char *argv[]) {
    int size;
    char *endptr;
    int datum;
    if (argc!=2) {
        fprintf(stderr,"Expected one argument\n");
        print_usage();
        exit(1); }
    size = strtol(argv[1], &endptr, 0);
    if (*endptr!=0) {
        fprintf(stderr,"Expected a well formed integer, found %s\n",argv[1]);
        print_usage();
        exit(1);
    }
    if (size0) {
        putc(0,stdout);
        size--;
    }
    return 0;
}
---- end of pad.c ----
```