

Instalando un BBS en Linux:

[Instalando un BBS en Linux:](#)

[1 INTRODUCCIÓN](#)

[2 MODOS DE ACCESO](#)

[2.1 ACCEDIENDO POR TELEFONO](#)

[2.1.1 Configurando el módem:](#)

[2.1.2 Conseguir que el modem conteste al teléfono:](#)

[2.1.3 ENLAZÁNDOLO TODO](#)

[2.2 Otra forma de acceder \(TCP/IP\)](#)

[2.2.1 Acceso por telnet](#)

[2.2.2 Protocolos específicos de transporte de FIDO vía IP](#)

[2.2.2.1 BinkP](#)

(puerto 24554)

[2.2.2.2 ifcico](#)

(puerto 60179, principalmente en sistemas unix)

[2.2.2.3 Vmodem](#)

(puerto 3141, soporta Telnet:23)

[2.2.2.4 Telnet](#)

(puerto 23)

[2.2.3 Acceso por HTTP](#)

[2.2.4 Acceso a las áreas de ficheros por FTP](#)

[2.2.5 Transporte del correo FIDO por Email y similares](#)

[2.2.6 Gates NEWS/FIDO](#)

Instalando un BBS en Linux:

No cabe duda de que Linux es uno de los sistemas operativos más adecuados y de mejor relación calidad/coste para cualquier aplicación que implique conexión a redes de transmisión de datos. Yo considero que también es un sistema operativo idóneo para instalar un BBS.

1 INTRODUCCIÓN

Este documento pretende cubrir ciertos aspectos particulares de cómo configurar una BBS en Linux. Es conveniente consultar también el SERIAL-HOWTO y el MODEM-HOWTO. Se hará un breve repaso por los paquetes de software.

2 MODOS DE ACCESO

Instalando un BBS en Linux:

Lo primero que debemos hacer para instalar nuestra BBS es decidir como van a acceder a ella los usuarios, ya que ello nos condicionará para decidirnos por el software que vamos a usar.

2.1 ACCEDIENDO POR TELEFONO

Desde que las redes de área extensa hicieron su aparición, la línea de teléfono convencional, y el módem, fueron uno de los fundamentales caminos de comunicación entre ordenadores, por ser algo que está al alcance de cualquiera. Antes del auge de Internet Redes de BBS como FidoNet o WWIV usaban este medio para conectarse entre ellas y con sus usuarios. Si decidimos que nuestro BBS tenga alguna conexión de este tipo, debemos conseguir primero que el módem funcione y atienda las llamadas. Si no podemos pasar al punto 2.2 directamente.

2.1.1 Configurando el módem:

Normalmente, nuestra BBS podrá ser accedida por red telefónica, para ello necesitaremos uno o varios módems, dependiendo de las líneas que tengamos, y deberemos configurarlos adecuadamente, tanto para recibir, como para realizar llamadas. Normalmente el módem estará conectado a un puerto serie.

Los puertos serie en Linux: Cada puerto serie en un PC debe tener una dirección de E/S, y una interrupción (IRQ).

Normalmente un PC viene equipado con dos puertos serie, pero puede tener cuatro o más. Estos son los cuatro puertos serie correspondientes a COM1 – COM4, con las direcciones e IRQ's habituales:

```
/dev/cua0, /dev/ttyS0 (COM1) dirección 0x3f8 IRQ 4
/dev/cua1, /dev/ttyS1 (COM2) dirección 0x2f8 IRQ 3
/dev/cua2, /dev/ttyS2 (COM3) dirección 0x3e8 IRQ 4
/dev/cua3, /dev/ttyS3 (COM4) dirección 0x2e8 IRQ 3
```

Si Linux no detecta ningún puerto serie cuando arranca, entonces asegúrese de que el soporte de comunicaciones serie está compilado y activo en el núcleo.

¿Por qué dos nombre para el mismo dispositivo (cua/ttyS)? Los dispositivos `/dev/ttyS*` son para conexiones de entrada y los dispositivos `/dev/cua*` son para conexiones de salida. Cuando nos referimos a entrada significa que pueden recibir llamadas. Esto es así, porque los dispositivos `/dev/cua*` se bloquean siempre que una aplicación los usa, los `/dev/ttyS*` por el contrario pueden ser usado por un daemon que compruebe continuamente si hay una llamada, sin que por ello se bloquee y permita a otras aplicaciones abrirlos para salida sin necesidad de detener el demonio. Generalmente podremos referirnos siempre a los `/dev/ttyS*` tanto para hacer llamadas como para recibirlas.

Nótese que por defecto estos dispositivos tienen IRQ's que se solapan. No se pueden compartir IRQ's si queremos usar los puertos `/dev/ttyS0` y `/dev/ttyS1` habremos de cambiarle la IRQ a uno de ellos.

Si optamos por usar una IRQ no estándar deberemos comprobar que está bien configurada utilizaremos el comando `setserial` para comprobar y ajustar los parámetros de los puertos serie:

```
setserial -a /dev/ttyS*
```

Nos muestra los parámetros configurados en el sistema para cada puerto.

```
setserial /dev/ttyS2 irq 5
```

Modificaría la configuración de `/dev/ttyS2` para usar la IRQ 5. Si necesitásemos modificar estos parámetros, es muy conveniente añadir la línea con las modificaciones necesarias a uno de los scripts de arranque por ejemplo el `rc.local`, para que se actualicen cada vez que arranca el sistema.

Si nuestro módem es externo, normalmente lo conectaremos a `/dev/ttys0` o `/dev/ttys1` y no tendremos mayor problema. Si es interno habrá que configurarlo para que use uno de los puertos `/dev/ttyS2` o `/dev/ttyS1` y o bien usamos una IRQ distinta a la estándar, o bien desactivamos el puerto que comparte IRQ con él.

Instalando un BBS en Linux:

Hay módems internos de los denominados winmodems que pueden ser mucho más complejos de configurar, son ligeramente más baratos, pero yo desaconsejo su uso en Linux.

Recientemente cada vez aparecen más módems USB para su correcta configuración os remito al USB-HOWTO. Solo resaltar que normalmente los puertos de módem USB se denominan: /dev/usb/ttyACM* y /dev/usb/cuaacm* para entrada y salida respectivamente.

Una vez configurado correctamente el puerto, de nuestro módem nos será muy útil establecer un enlace simbólico del dispositivo /dev/modem al puerto al que está conectado, así todas las aplicaciones hablarán con /dev/modem, y si un día cambiamos el modem de sitio solo hay que cambiar el enlace, y no reconfigurar todas las aplicaciones, por ejemplo suponiendo que nuestro modem está en el segundo puerto serie haremos.

```
ln -s /dev/ttyS1 /dev/modem
```

Si tenemos varios módems podemos hacer varios enlaces tales como /dev/modem1 /dev/modem2 etc...

Para asegurarnos de que todo está bien lanzaremos el minicom o cualquier otro programa de comunicaciones y configurándolo para el dispositivo /dev/modem podríamos probar que lanzando un AT el modem nos contesta con un OK o algo similar.

2.1.2 Conseguir que el modem conteste al teléfono:

Una vez que el módem está debidamente configurado, solo nos queda conseguir que conteste al teléfono. La tarea no es complicada, pero en Linux, con respecto a sistemas operativos de tipo DOS, la cosa es bien distinta. En las BBS's que conocemos de DOS era el propio software de la BBS el que continuamente supervisa si hay una llamada, y contesta. En Linux podría hacerse así, pero yo aconsejo usar las herramientas propias del sistema. En Linux el programa que permite que un terminal se conecte al sistema, para dar paso al login, se denomina getty, existen diversas variaciones que añaden ciertas características, concretamente para uso con módems disponemos de mgetty, que puede gestionar todo lo relativo a una conexión por módem, discriminar si la llamada es de fax o de módem (e incluso de voz si el módem tiene esta capacidad), y si es de módem puede distinguir si llamamos con protocolo PPP, con un mailer FTN o con un simple programa de terminal. Una vez determinado el tipo de llamada pasará el control de ésta al programa que le digamos.

Comprobaremos que tenemos instalado mgetty, y si no es así lo instalamos, viene en casi todas las distribuciones, pero por si no fuera el caso puede encontrarse fácilmente en la red, concretamente aconsejo la instalación de un paquete llamado mgetty+sendfax que nos permite además gestionar el envío y recepción de faxes.

Normalmente mgetty es lanzado por el propio init, al igual que todos los demás getty's que controlan los terminales que tengamos en el sistema, así que lo primero que deberemos hacer es modificar el /etc/inittab para indicarle que debe lanzar mgetty, y lo haremos incluyendo una línea similar a esta por cada módem que tengamos contestando al teléfono:

```
S1:2345:respawn:/usr/sbin/mgetty /dev/ttyS1
```

Describo cada uno de los campos de la línea, y lo que significan:

S1 es una etiqueta, y lo único que tiene que cumplir es que no se repita, si tuviésemos varios módems usaríamos S1, S2, S3... para cada línea.

2345 son los RUNLEVEL afectados, quiere esto decir que el mgetty para este puerto se lanzará en siempre que arranquemos nuestro PC en cualquier RUNLEVEL del 2 al 5.

/usr/sbin/mgetty es el programa que lanzamos, normalmente se instala en ese directorio, pero asegurados por si acaso, de donde se encuentra en vuestra distribución concreta, y lo que va a continuación son los parámetros hay que poner obligatoriamente el puerto que controla, y pueden incluirse también otras opciones, yo por ejemplo le pongo -n 2 para que conteste al segundo timbre, ya que tengo servicio de identificación de llamadas, y si contestas al primer timbre da problemas de conexión (el número que llama se transmite entre el primer y segundo timbre, si el módem descuelga en ese momento puede interpretar esos datos como que vienen del módem que llama y no establecer bien la conexión).

```
S1:2345:respawn:/usr/sbin/mgetty -n 2 /dev/ttyS1
```

Sería muy interesante ahora editar el fichero de configuración de mgetty que normalmente se encuentra en

Instalando un BBS en Linux:

/etc/mgetty+sendfax/mgetty.config

Un ejemplo puede ser este:

```
#####
# Extracto de /etc/mgetty+sendfax/mgetty.config
#
# ----- global section -----
speed 115200
#
# ----- port specific section -----
#
# Puerto serie donde tenemos el módem
port ttyS1
#
# Nivel de depuración.
debug 9
#
# Ponerlo a yes si queremos recibir faxes
data-only no
#
# Detección automática del tipo de módem (voz, módem-fax, etc...)
modem-type auto
#
# Numero de timbres que suenan antes de que mgetty responda
rings 1
#
# En la secuencia de inicialización del módem, hay que desactivar la
# 'respuesta automática'. Es mgetty quien descuelga, no el módem.
# Normalmente se consigue incluyendo S0=0. Ajustar el init-caht
# de acuerdo con nuestro modem.
#
init-chat "" ATZ OK ATH0 OK ATSO=0 OK
# -----<end ttyS1>----
```

Ahora nos queda configurar el login: en función de cómo se discrimine la llamada mgetty decidirá que programa va a ejecutar a continuación según las reglas establecidas en el fichero /etc/mgetty+sendfax/login.config. Se puede lanzar

```
    ifcico      (FidoNet)
    o uucico    (UUCP)
    o pppd      (ppp)
    o telnet    (telnet)
    o login     (terminal)
    o etc...
```

```
#/etc/mgetty+sendfax/login.config
#
#
# el login_program es el programa que se ejecutará, con los
# argumentos pasados en [arguments]. Una "@" en los argumentos
# será sustituida por el nombre de usuario que hizo el login.
# AVISO: si no se usa "@" el programa login no puede saber
# el nombre del usuario que acaba de acceder.
#
#-----
#
# UUCP
# ----
# Esta línea indica a mgetty que si un login comienza por U
# se trata de una conexión UUCP y por tanto debe lanzar el
# uucico (o el programa que atiende UUCP)
#
U*      uucp    @          /usr/lib/uucp/uucico -l -u @
#
# FIDONET
# -----
# Si el login es la palabra /FIDO/ lanzaremos el programa
# que atiende las llamadas de otro sistema FTN, ifcico,
# qico, mbcico son ejemplos de mailers que pueden realizar
# esta tarea.
```

Instalando un BBS en Linux:

```
#
#
# FIDO/ uucp      fido      /usr/local/lib/fnet/ifcico @
#
# Si el login el /AutoPPP/ se lanzará el pppd para atender la
# llamada procedente de una conexión TCP/IP
#

/AutoPPP/ - a_ppp /usr/sbin/pppd auth -chap +pap login debug

#
# EL RESTO DE USUARIOS ENTRARAN EN EL LOGIN NORMAL
# -----
#
# *      -      -      /bin/login @
#-----<EOF>----
```

Ahora para probar que todo funciona debemos reiniciar el proceso init para que vuelva a leer la configuración del inittab y arranque el/los mgetty(s) configurados:

Kill -HUP 1

Probaremos a hacer una llamadas y comprobar si el módem responde correctamente.

2.1.3 ENLAZÁNDOLO TODO

De todo lo explicado anteriormente, hay que hacer excepciones, me he puesto en el caso más complicado de configurar, que es que tengamos un programa distinto para atender cada tipo de conexión entrante, y las comunicaciones en sí sean manejadas por el propio sistema operativo. Existen excepciones a esta norma, y en algún paquete de software de BBS como BBBS por ejemplo, no es necesario todo esto, ya que el propio software de la BBS atiende al puerto de comunicaciones y es capaz de gestionar por sí mismo de que tipo de conexión se trata y actuar en consecuencia, al estilo de las BBSs para DOS.

Aclarada esa excepción continúo explicando como haremos en el resto de los casos:

Si llama un Nodo o un Punto con un mailer FTN el mgetty lo detectará, y transferirá el control al Mailer que le indiquemos en el login.config.

Si llama un usuario en una conexión de acceso telefónico a redes con protocolo punto a punto (PPP) también el mgetty lo detectará y transferirá el control al pppd, que atenderá esta conexión.

Lo normal en las BBS's para Linux es que se puedan configurar para actuar como si fuesen una shell, algunas mantienen su propia base de usuarios, y no usan para nada las cuentas de usuario en nuestra máquina. Otras usan el sistema de autenticación de Linux y guardan los datos de usuario y contraseña en /etc/password, y cada usuario tiene su directorio home, donde se guarda el resto de opciones de configuración. Otras usan un método combinado, y cada usuario tiene una entrada en nuestro /etc/password donde se almacena el nombre de login y la contraseña, y luego tienen su propia base de datos con el resto de los datos.

En el primer caso lo normal es crear un usuario con el nombre 'bbs', sin password y ponerle como shell el programa de la BBS que atiende las llamadas de terminal. En el fichero issue, que se muestra antes del login, informaremos al usuario que escriba 'bbs' cuando se le pida login. En cualquiera de los otros casos la propia documentación de cada BBS nos indicará como es mejor actuar. En cualquier caso esto es una gran ventaja, ya que al disponer del auténtico login de nuestra máquina Linux en el puerto de comunicaciones, podremos desde cualquier sitio conectarnos por módem para administrar la máquina.

2.2 Otra forma de acceder (TCP/IP)

Actualmente, el acceso por teléfono, que sigue siendo un método válido y al alcance de cualquiera, puede ser caro en caso de conexiones de larga distancia, es por ello que dado el abaratamiento de Internet cada vez se utilice más esta red como transporte en el mundo de los BBSs. Hay muy diversas maneras de integrar las BBS y Fido en Redes de tipo

Instalando un BBS en Linux:

TCP/IP.

2.2.1 Acceso por telnet

Si nuestra BBS está actuando como shell, no hay que hacer nada, más que instalar y configurar un servidor de telnet en nuestra máquina, para poder acceder a ella por telnet. Otros sistemas, como BBBS, tienen su propio daemon que actúa como servidor de telnet, http y FTP, así que solo es cuestión de configurarlos según indique su documentación.

2.2.2 Protocolos específicos de transporte de FIDO vía IP

Nos referimos con esto a establecer una conexión entre dos programas compatibles FTN (dos mailers), pero usando como transporte la red Internet o una red TCP/IP, en lugar de una conexión directa por teléfono. Evidentemente al ser el medio distinto son necesarias ciertas variaciones en el protocolo de conexión.

En estos momentos son varios los protocolos que se utilizan para establecer una sesión FTN sobre IP.

2.2.2.1 BinkP

(puerto 24554)

Usado inicialmente por el mailer sobre IP BinkD, disponible para muchos sistemas operativos Free BSD, Linux, OS/2, Win95, Win-NT, Amiga... Tanto las especificaciones del protocolo BinkP como el código fuente del propio BinkD son públicos por lo que el protocolo actualmente está integrado en varios programas disponibles para diversos sistemas operativos: Argus (Windows) Beemail (Windows, OS/2), MbseBBS (Linux), BBBS (Linux, OS/2, Windows)...

2.2.2.2 ifcico

(puerto 60179, principalmente en sistemas unix)

ifcico es el mailer usado por el paquete ifmail, está disponible para Linux y Unix. Ifcico usan protocolo específico incompatible con otros mailers, actualmente existen variaciones sobre la versión original que pueden comunicarse mediante el protocolo Telnet estándar que es soportado por la mayoría. Consultar las páginas de Christof Meerwald que coordina la versión llamada ifcico-cm.

2.2.2.3 Vmodem

(puerto 3141, soporta Telnet:23)

Vmodem era inicialmente un controlador de dispositivo para OS/2 que simulaba un módem sobre una conexión telnet, hoy están disponible en muchos sistemas operativos, programas como com/ip para Windows o modemu para Linux nos permiten que un programa inicialmente pensado para trabajar con una conexión directa por módem utilice en su lugar una conexión telnet. Normalmente estos programas se comunican perfectamente con el Vmodem de OS/2.

2.2.2.4 Telnet

(puerto 23)

Conexión genérica por telnet soportada por la mayoría de los programas.

En la nodelist se han añadido una serie de Flags para indicar la capacidad de cada nodo para aceptar conexiones mediante estos protocolos. Consultar los FTS-5000 y FTS-5001 para más detalles.

2.2.3 Acceso por HTTP

Instalando un BBS en Linux:

2.2.4 Acceso a las áreas de ficheros por FTP

2.2.5 Transporte del correo FIDO por Email y similares

2.2.6 Gates NEWS/FIDO