

Servidor de Acceso telefónico a Redes



Orlando Alemán Ortiz
Samuel Díaz Cabrera

4º Ing. Informática
Curso 2005/06

Licencia



Esta obra ha sido publicada bajo licencia "Reconocimiento-NoComercial-CompartirIgual 2.5 Spain" de Creative Commons, la cual implica que:

Usted es libre de:

- copiar, distribuir y comunicar públicamente la obra
- hacer obras derivadas

Bajo las condiciones siguientes:



Reconocimiento. Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador.



No comercial. No puede utilizar esta obra para fines comerciales.



Compartir bajo la misma licencia. Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.

Y además:

- Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra.
- Alguna de estas condiciones puede no aplicarse si se obtiene el permiso del titular de los derechos de autor
- Los derechos derivados de usos legítimos u otras limitaciones reconocidas por ley no se ven afectados por lo anterior.

Para ver una copia de esta licencia, visite <http://creativecommons.org/licenses/by-nc-sa/2.5/es/> o envíe una carta a Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Índice de contenidos

1. Previos.....	2
Introducción.....	2
Equipamiento empleado.....	2
2. Desarrollo.....	3
Instalación del software.....	3
Configuración.....	3
MGeTTY.....	3
PPPd.....	5
Cambios a nivel de sistema.....	7
3. Conclusiones.....	9
4. Referencias.....	10

1. Previos

Introducción

En esta quinta práctica avanzaremos en el tema de conexión *Punto-a-Punto* entre dos máquinas, abandonando el lado del cliente para centrarnos en la configuración y puesta en funcionamiento del servidor de acceso.

Como siempre, tendremos en cuenta tanto a Windows como a Linux del lado del cliente, pese a que en el lado del servidor ejecutaremos exclusivamente *GNU Linux*.

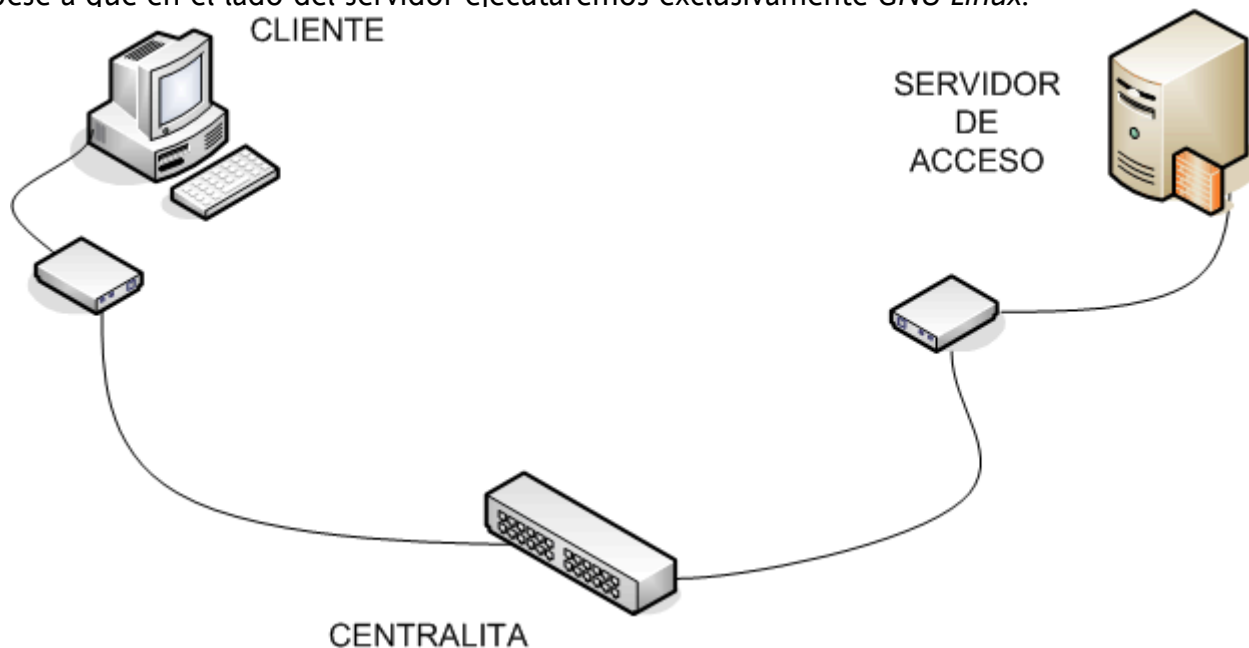


Ilustración 1: Conexión Punto a Punto (PPP)

Equipamiento empleado

Hardware:

- 2 x (PC + Módem v9X compatible Hayes) [preinstalado]
- 1 x Centralita telefónica por pulsos
- 1 x tarjeta de ethernet + cable RJ45 [servidor, preinstalado]
- Cableado RJ11 + Rosetas RJ11 [preinstalado]

Software:

- *Fedora Linux* y *Fedora Linux* del lado del cliente, y *Fedora Linux* del lado del servidor.
- En las instalaciones de *Fedora*, “*pppd*” y “*chat*” si se va a actuar como cliente o “*mgetty*” y “*pppd*” si se va a actuar como servidor.
- Driver modem [ya instalado]

2. Desarrollo

En este apartado explicaremos cómo montar un servidor de acceso telefónico a redes sobre GNU Linux utilizando la combinación software de “*mgetty*” con “*pppd*”. Para ello procederemos primeramente a la instalación de las aplicaciones, para luego pasar a configurarlas.

Instalación del software

Para instalar “*mgetty*” hacemos uso de la herramienta de gestión de paquetes “*yum*”, que nos permite buscar y descargar el software pedido desde un repositorio web, así como resolver sus posibles dependencias. En nuestro caso,

```
$ yum install mgetty mgetty-sendfax
```

Como método alternativo se podría realizar una instalación manual, haciendo uso del comando “*rpm*”. Aunque, en este caso, tanto la resolución de dependencias como la obtención de los paquetes pasarían a ser de nuestra responsabilidad. La sintaxis para instalar un paquete RPM es

```
$ rpm -ihv [paquete(s)]
```

Configuración

Una vez nuestro sistema dispone de los componentes necesarios para el funcionamiento del servidor, pasamos a su configuración e integración con el resto del entorno software.

MGeTTY

En primer lugar debemos establecer los parámetros adecuados para que “*mgetty*” pueda realizar correctamente la gestión de llamadas entrantes.

Los ficheros de configuración de “*mgetty*” se encuentran bajo el directorio “*/etc/mgetty+sendfax*”. Los más relevantes para nuestro objetivo actual son “*mgetty.config*” y “*login.config*”. El primero nos permite fijar el comportamiento de “*mgetty*”; mientras que el segundo determina las opciones de autenticación y acción a llevar a cabo una vez se haya establecido un vínculo físico.

“*mgetty.config*”

El elemento básico de definición en este fichero es el puerto. Un puerto, aquí llamado “*port*”, permite establecer las opciones de configuración particulares que debe utilizar el servidor cuando realiza una conexión sobre un dispositivo concreto. Esto no implica, sin embargo, que no puedan establecerse parámetros globales que sean comunes a todos los dispositivos a través de los cuales “*mgetty*” acepte llamadas. La forma de hacerlo, es simplemente añadir la línea de configuración al comienzo del fichero, justo antes de

definir cualquier sección “*port*”.

Todo puerto debe corresponderse con un dispositivo del sistema. Así, si por ejemplo un módem se encontrara en el primer puerto serial tendría por identificador de puerto “*ttyS0*”. En nuestro caso, dada la peculiaridad del módem del que disponemos (*softmodem HSF*), el puerto a ser configurado será el “*ttySHSF0*”.

Para nuestros objetivos, el siguiente fragmento de definiciones es más que suficiente:

```
# Nivel de logging máximo
debug 9
# Velocidad de acceso al módem (DTE): 38400 bps
speed 38400
# Inicio Zona de Definición de puertos
port ttySHSF0
  modem-type auto
  init-chat "" AT&F1 OK
  speed 230400
  rings 2
  data-only yes
```

De acuerdo a lo que hemos afirmado con anterioridad, “*debug*” y “*speed*” serán parámetros globales a utilizar por todos los dispositivos. “*debug*” es una opción que, si se especifica, fija el nivel de depuración a utilizar al generar el fichero de logs del demonio (en “*/var/log/mgetty.log.[puerto]*”). El desarrollador de “*mgetty*” opina en su documentación que un buen valor sería “4”, aunque es posible incrementar el nivel de detalle hasta “9”. “*speed*”, en cambio, decide la velocidad de acceso al módem (DTE). Su valor por defecto viene indicado por el valor que en tiempo de compilación se fijó como `DEFAULT_PORTSPEED`. En nuestro caso, nos basta con una velocidad que acepte el dispositivo, como es “38400”.

Con la primera entrada (“*modem-type auto*”) en la sección para “*ttySHSF0*” indicamos al demonio “*mgetty*” que autodetecte el tipo de módem que se está utilizando.

“*init-chat*” establece la secuencia que debe utilizarse para inicializar el módem. En la mayoría de los casos, y éste no será una excepción, la secuencia comienza por “””, para indicar que no debe esperarse nada para ejecutar el primer comando. “*AT&F1*” resetea el módem para dejarlo en un estado inicial dado por el perfil de fábrica 1.

“*rings 2*” ordena a “*mgetty*” responder a las llamadas entrantes tras 2 tonos.

“*speed 230400*” indica que la velocidad de transmisión a utilizar en este puerto será diferente a la global. En este caso, fijaremos la máxima posible.

Y finalmente, con “*data-only yes*” señalamos que sólo puedan realizarse transmisiones de datos.

“options.servidor”

Cada línea de este fichero determina una configuración para llevar cabo la identificación del usuario que desea la conexión y tiene el formato siguiente:

```
username userid utmp_entry login_program [arguments]
```

Cuando haya una llamada, “*mgetty*” se hará cargo de ella identificando al usuario entrante, reconociéndolo (viendo si aparece en “/etc/passwd”) y creándole una sesión temporal en el sistema. Seguidamente le solicitará autenticación por medio de un software de logueo, que generalmente es “/bin/login”.

En nuestro caso, el interés es únicamente el de arrancar el protocolo PPP nada más se reciba una petición de configuración de enlace (*LCP*). No nos interesa ninguna solicitud de identificación a este nivel. “*Automatic PPP*” es la solución a esta inquietud y constituye una opción que debe ser habilitada en tiempo de compilación. Dado que la mayoría de usuarios suelen requerir esta funcionalidad, suele estar (y lo es en este caso) activada en los paquetes finales.

Para usar “*Automatic PPP*” debe añadirse una línea como la siguiente:

```
/AutoPPP/ - a_ppp /usr/sbin/pppd file /etc/ppp/options.servidor
```

,donde “*pppd*” será el programa que controlará el protocolo de comunicaciones que actuará sobre el enlace y su argumento “file” indica a “*pppd*” que tome por fichero de configuración “/etc/ppp/options.servidor” en vez del habitual “/etc/ppp/options”. “*a_ppp*” será el usuario temporal para la sesión que habrá creado “*mgetty*”.

PPPd

En este apartado abordaremos los aspectos de configuración básicos para el uso de “*pppd*” como gestor del protocolo PPP del lado del servidor. Los ficheros que vamos a preparar deben ir bajo “/etc/ppp”.

“options.servidor”

Es el archivo en el que almacenaremos los parámetros que “*pppd*” necesita en su ejecución para satisfacer nuestras necesidades. Surge como alternativa a “options”, el cual dejamos para configuraciones cliente. La diferencia más importante entre usar “*pppd*” en modo servidor o en modo cliente radica en que, en el primero, entrega una IP al cliente y lo autentifica, y en el segundo es él el destinatario de esas transacciones.

Las opciones que utilizaremos vienen recogidas y comentadas en la siguiente tabla:

Término	Explicación
Configuración base	
-detach	Desasocia PPPd del terminal
crtcts	Control flujo hardware
auth	Obliga al otro extremo a autenticarse
login	Cuando usamos "login" con PAP, los usuarios declarados PAP sin especificar una contraseña, tendrán por clave la de la base de contraseñas del sistema. Esto implica que deben haber sido declarados como usuarios locales al servidor.
require-pap	Solicita autenticación PAP
refuse-chap	Rechaza autenticación CHAP
asynmap 0	NO establecer ninguna secuencia de control en forma de cadena de escape
modem	Usar las líneas de control del módem [Opción por defecto]
proxyarp	Crea una entrada ARP para el otro extremo asociada la dirección física del servidor, haciendo el efecto de que el cliente está en la red del servidor
idle 1000	Indica el tiempo, en milisegundos, de inactividad máximo
maxconnect 120	Indica el tiempo máximo, en segundos, de una sesión
Configuración Interred	
172.16.3.1:172.16.3.2	[IPServidor]:[IPCliente]
netmask 255.255.255.0	Suministra la máscara
ms-dns 193.145.147.22	DNS1
ms-dns 193.145.143.100	DNS2

De todas estas sentencias, sólo las dos últimas (“*ms-dns*”) son inútiles en entornos *GNU Linux*, ya que únicamente suministran valores a sistemas compatibles en cuanto a interpretación del protocolo con *Microsoft Windows*. Esto implica que si el cliente trabaja sobre una plataforma *Unix* deba precisar las *DNS* manualmente (añadiéndola a “/etc/resolv.conf”).

“pap-secrets” y “chap-secrets”

Los ficheros de configuración PAP y CHAP son respectivamente “/etc/ppp/pap-secrets” y “/etc/ppp/chap-secrets”. Son utilizados por “pppd” durante la fase de autenticación; es decir, justo después de que “pppd” sea invocado desde el demonio “mgetty”. Nótese que la fase de autenticación tendrá lugar debido a que hemos establecido la opción “auth” en el fichero anterior.

PAP consiste básicamente en el envío por parte del cliente de un nombre de usuario y una contraseña al servidor sin aplicarle ninguna técnica de encriptación avanzada. Por lo que es vulnerable a intrusos que puedan intentar obtener la contraseña escuchando de la línea serie o mediante otros métodos.

CHAP, en cambio, no presenta esta inseguridad. El servidor envía una ristra de caracteres generada aleatoriamente al cliente, junto a su nombre de *host*. El cliente con esta información genera una clave encriptada, que envía al servidor junto a su nombre de

equipo. Es entonces cuando el servidor hace la misma comprobación y responde al cliente si es correcta o no la validación. Además CHAP realiza esta comprobación a intervalos regulares para asegurarse de que el cliente no ha sido desconectado o reemplazado por un intruso.

Para nuestros propósitos, la autenticación PAP es más que suficiente. Así que centraremos nuestra atención únicamente sobre “pap-secrets”. Cada usuario de PPP debe disponer de una entrada él. Una entrada es la composición de cuatro campos delimitados por espacios (o tabuladores) de la forma:

```
user      server    password  acceptable_IPs
```

El significado de estos parámetros viene indicado en la siguiente tabla:

Campo	Explicación
user	Identificador de usuario, expresado entre comillas
server	La opción de nombre de servidor no es requerida en los archivos de configuración PPP. Se pone * para indicar que host es válido.
password	Clave del usuario, escrita entre comillas
acceptable_IPs	Indica las direcciones IP desde las que se debe haber hecho la llamada. Con * se aceptará cualquier dirección IP.

El fichero que hemos elaborado contiene las siguientes líneas:

```
# Secrets for authentication using PAP
# user      server    password  acceptable_IPs
"prueba"   *          "prueba"  *
"redes"    *          ""         *
```

En la primera, se autoriza al usuario “prueba” a acceder con la contraseña “prueba”, constituyendo un caso de asignación de contraseña explícito. La segunda entrada, en cambio, requiere que el usuario exista también en el sistema, de donde se leerá la contraseña. Ésto es posible porque hemos incluido la opción “login” entre las opciones de configuración.

Cambios a nivel de sistema

El primer cambio a realizar es informar al sistema de que debe atenderse un nuevo terminal, el del demonio “mgetty”, para lo que basta con incluir una nueva entrada a “/etc/inittab”.

“inittab” es el fichero a partir del cual arranca la ejecución de “init”, el proceso padre de todo sistema GNU Linux. Dirige y supervisa la ejecución de los demonios, terminales y demás procesos del sistema. En el caso de “mgetty” posibilitará que la aplicación sea ejecutada como demonio al arrancar y que tras una desconexión se reinicialice.

Cada entrada en el fichero está compuesta por cuatro valores separados por el carácter “:”. El primero, es una etiqueta identificadora; el segundo, una lista con los niveles de ejecución donde debe ejecutarse el demonio; el tercero, describe cuándo hay que realizar la acción detallada en el cuarto campo y, finalmente, el cuarto campo indica la acción a realizar.

```
# Sintaxis de una entrada en inittab  
[id]:[runlevels]:[acción]:[proceso]
```

La inclusión a realizar es la siguiente:

```
# Sintaxis de una entrada en inittab  
S0:2345:respawn:/sbin/mgetty ttySHSF0
```

Es decir, “*mgetty*” debe iniciarse en los niveles de ejecución 2, 3, 4 y 5 utilizando como puerto de escucha el “*ttySHSF0*”. La opción “*respawn*” hace que “*init*” vuelva a ejecutar el demonio tras una desconexión, asegurando la disponibilidad del servicio.

Una vez editado el fichero “*inittab*” debemos reiniciar el proceso “*init*”, lo cual conseguimos ejecutando “*kill -1 1*”.

Si queremos proporcionar *Internet* debemos tener configuradas correctamente las interfaces de red, de forma que se disponga de acceso hacia el exterior, y, además, activar el enrutamiento *IP*.

Para aprender a configurar la Red y activar el enrutamiento *IP* remitimos al lector a la memoria de la práctica 2.

Con estas configuraciones, podremos ya probar el correcto funcionamiento del servidor tras la labor realizada. El fichero “*/var/log/mgetty.log.ttySHSF0*” será testigo de todos los eventos que se produzcan en el sistema. Podemos leerlo en cualquier instante con cualquier editor de textos, aunque la mejor idea sea utilizar la orden “*tail -f [fichero]*”.

3. Conclusiones

La instalación manual de un servidor de acceso telefónico a redes en *GNU Linux* es una tarea ardua y llena de dificultades, por la alta cantidad de factores (opciones, disponibilidad de la línea, etc.) que pueden influir sobre la conexión. Pero el resultado final es gratificante.

“*pppd*” es una herramienta realmente flexible, que puede realizar funciones tanto en el cliente como en el servidor en combinación con un software que gestione el enlace a nivel físico.

4. Referencias

- “*Linux PPP Howto*” Corwin Light-Williams, Joshua Drake. Proyecto Lucas.
URL: <http://www.tldp.org/HOWTO/PPP-HOWTO/>
- “*Guía de Linux (Ambiente Global)*”. Apartado: “*Dial-up Configuración Módem PPP*”
URL: <http://www.osmosislatina.com/linux/index.jsp>
- “*Man pages*” Apartados: pppd, chat, mgetty
URL: <http://www.die.net/doc/linux/man/>
- “*Linux Dialin Server Setup Guide*”. Josh Gentry
URL: <http://www.swcp.com/~jgentry/pers.html>
- “*mgetty+sendfax*”. Gert Doering
URL: <http://mgetty.greenie.net>