Bisoños Usuarios de Linux de Mallorca y Alrededores | Bergantells Usuaris de Linux de Mallorca i Afegitons

Configurando el router 3com 812 con pppoe para ADSL (IP dinámica / Telefónica) bajo linux.

Por **alex**, ***z0mbie*** (http://sumidero.host.sk/)
Creado el 12/05/2003 19:48 y modificado por última vez el 12/05/2003 19:48

En este artículo trataré de explicar algo, que hasta ahora, no estaba muy documentado (al menos yo no he encontrado casi nada al respecto y tampoco ha sabido contestarme nadie, ni en foros ni en listas de correo) que es el cómo configurar un router 3com 812 OfficeConnect (el típico que pone Telefónica para las ADSL) bajo linux de tal manera que actue basicamente como un modem, ¿ el porqué ? será algo que intentaré analizar a lo largo de este documento.

Este es mi primer artículo en bulma y me he liado un poco con lo del formato :−P, en fin, a ver si no sale muy fozado.

No se como tendra la mayoría configurado su router o como vendrán defábrica, pero yo inicialmente tenía una configuración tal que:

- **IP interna** del router: 172.26.0.1 (por aquí se podía acceder a los servicios internos propios del modem: telnetd, httpd y tfptd).
- **IP publica** del router: dinámica.
- **dpat** (default port address translation) al host 172.26.0.2
- **filtros** que denegaban el acceso desde internet a algunos puertos que podríamos considerar "peligrosos".
- un perfil por defecto que segun encendías el router levantaba la conexión automaticamente mediante el propio *pppoe* interno del router.

Cuando contraté el servicio de ADSL, ingenuo de mi, pense que por solicitar un router, todo funcionaría a las mil maravillas bajo linux y ya tenía en mente un cuento de la lechera sobre la instalación de un servidor web que podría permanecer encendido las 24 horas así como alguna otra cosa. No tardó mucho en romperse el cantaro de la leche, pero bueno, como no hay mal que cien años dure o gente como yo que en ausencia de algo mejor que hacer se rompe el coco con estas cosas, llego la solución 8−).

Como dato informativo, por lo que me han contado y he leído, la mayoría de la gente con un contrato como el mio o similar (IP dinámica, porque la fija tiene una tarifa mensual relativamente superior) que tiene router, lo usa en una configuración **multipuesto**, es decir, el router se encarga de hacer el *nat* y de *filtrar* desde internet hacia la red local. Esto significa, que es también el propio router quién tiene la *IP pública* y nuestro equipo, o nuestros (si tenemos una red) tienen las privadas. A mi, personalmente, esto no me acababa de convencer, por varias razones que expongo a continuación:

- Lo de filtrar (*firewall*) a nivel router, no me ofrecía mucha versatilidad ni comodidad. Para cambiar algo había que entrar al router, y yo soy de esos que cuando quiere abrir el ftpd solo durante un momento para que alguien suba/baje algo, me gusta tener el control en la línea de comandos con una orden para el ipchains/iptables.
- Me gusta saber en tiempo real por medio de logs lo que va pasando por la red (intentos de conexión a puertos y todo eso) a través de una consola coloreada y si, ya se que el propio router tiene una opción para enviar a un *syslog remoto* información pero la he probado y desde luego no tiene ni punto de comparación con lo que yo deseaba.Para los curiosos, uso el ippl[1] junto con **iptables** para loggear y el ccze[2] para verlo todo mas bonito y coloreado.
- Por estas 2 razones anteriores, no se me ocurrió otra cosa que efecutar una chapuza semejante a:

dpat **nat**

internet <−−−> router 3com <−−−> router linux <−−−> red local

172.26.0.1	172.26.0.2	192.168.1.x
IP dinámica	192.168.1.1	

El router 3com hacía **dpat** (default port address) al host 172.26.0.2 (vamos, que redireccionaba todo lo que entraba por defecto del router 3com al router linux) y luego este volvía a hacer **nat** a la red 192.168.1.0. Esto me permitió no tener que modificar las antiguas configuraciones que tenía del firewall de cuando usaba la conexión por modem. Pronto descubrí, que además de ser la peor organización de red de toda la historia, *murphy se sacaba problemas* de la manga cada dos por tres. Los más típicos, **protocolos que abrían sockets a puertos que el nat no era capaz de resolver** (y no era problema del nat de linux, ya que antes, con el modem, funcionaban perfectamente), como puedan ser el puerto de datos del ftp, DCC del irc y otros. Hasta el router linux llegaban y la cosa funcionaba sin problemas, pero en cuanto intentaba algo desde los equipos de la red 192.268.1.0 no había manera. Además, no se si derivado de esto, en los logs, no paraban de verse mensajes de *checksum error en los nat del kernel*, cosa que tampoco me parecía óptima.

- Por supuesto, como cualquier hijo de mortal, al no tener una IP fija, tenía pensado utilizar un *dominio dinámico*. Aquí hubo una coña durante los días que tuve montado esta configuración anterior, ya que generalmente a los clientes que hacen el update del dominio (en mi caso ddclient[3]) hay que pasarles como parámetro la IP pública de internet y claro, el único que la conocía era el router. En un principio pensé algo tan grotesco como hacer un script en Perl que conectara por telnet al router, listara la configuración y filtrara la IP pública de la red wan. ¿ Y porqué todo este lío pensaran algunos ? pues porque averiguar la IP remota, cosa que generalmente hacía hasta ahora, entrando en una web que me lo dijera, al usar el *proxy de Telefónica* no es de gran utilidad.
  Luego descubrí que hay una web[4] que te da ambas IPs, la del proxy y la original. Os pongo a continuación como curiosidad lo que usaba para obtener la IP pública que posteriormente pasaría como parametro del ddclient:

  **echo −en "GET http://www.showmyip.com/ HTTP/1.0\r\n\r\n" | **
  **nc www.showmyip.com 80 | grep "TITLE" | **
  sed −e 's/.*Connection //' | sed −e 's/ −−>.*//'
  (dependiendo del proxy, puede funcionaros o no la expresion regular del sed)
- Luego, estaba el tema, de que en el futuro tenía pensando implementar algo que equilibrase el ancho de banda para que los programas de leech y p2p no me consumieran todo en beneficio de otros como el correo, ssh o similares, cosa totalmente inviable con el router.

En principio, todo esto que yo quería era totalmente factible y a todo el mundo parecía irle bien con una **configuración monopuesto e IP fija**. En este tipo de configuraciones, segun tengo entendido, la gente le pone a linux la IP fija y al router le pone una IP que calcula haciendo una operacion AND sobre la IP fija. A mi por supuesto, esto no me valía de nada, ya que cada vez que conectaba, me ofrecían una IP diferente, así que empecé a leer manuales, investigar y todo eso :−)

A continuación y sin enrrollarme más, los requisitos para hacer rular el tema:

- **Kernel 2.4.x**. Por ahora estoy usando el **bf24** de Debian[5] woody. Al principio estaba haciendo las pruebas con uno de la rama *2.2.x*, que era lo que había usado desde siempre en el equipo del servidor y el pppoe no me rulaba ni para atrás. Un día me dio por arrancar con la knoppix[6] y probar con esta, cual fue mi sorpresa al ver que iba. Empece a descartar cosas como pudieran ser versiones de ppp, pppoe y al final descubrí que era el kernel.
- **Un router 3com 812 OfficeConnect** (en otros, supongo que la configuración de bridge varía totalmente) con **firmware V2.0.0** (con otras revisiones, es posible que cambien algunos comandos, tendreis que consultar el manual del CLI)
- **Paquetes**: pppoe, pppd, pppoeconf (opcional), minicom (con un cable serie para configurar el router).
- Debian unstable/sid: bueno, esto lo pongo a modo informativo, ya que aunque debería funcionar con otras distribuciones o ramas de Debian, yo lo he hecho sobre esta y con las versiones de los paquetes que ello implica, con lo que es la única en la que aseguro el funcionamiento. Bueno, como dije antes, con la knoppix tambien me fue 8−)
- **1 tarjeta de red** para conectar con el router, o 2 si deseamos hacer *nat* a una red interna. No estoy seguro de si podríamos usar una única (por medio de *ip aliasing* de la misma manera que se consiguen usar 2 ips con la misma tarjeta) con la configuración del bridge, ya que me parece rizar el rizo. En fin, si alguien prueba y lo consigue, que diga algo :−)

Pasos a seguir:

- Antes de poner en peligro nuestra "conexión" con internet, bajarse todo el software necesario en caso de no tenerlo en cds.
- Entramos en el router por medio del puerto serie, usando **minicom**, para usarlo *debemos tener permisos sobre el /dev/ttySx* pertinente. Una vez que hayamos cargado minicom habrá que configurarlo con estos parámetros (se entra en la configuración con **Crtl+A y luego O**):
  En la sección **"Serial port setup"**:

  A − Serial device : /dev/ttyS0 (si usais otro, el que sea)
  E − Bps/Par/Bits : 9600 8N1
  F − Hardware Flow Control : Yes
  G − Software Flow Control : No

  En la seccion **"Modem and dialing"**:

  A − Init string : borramos lo que haya y lo dejamos en blanco
  B − Reset string : borramos lo que haya y lo dejamos en blanco

  Grabamos la configuración y si todo ha salido bien debemos ver el *prompt del router* (**3Com−DSL>**), en caso contrario, con pulsar enter una o dos veces, suele aparecer.

  Procedemos a *borrar la configuración actual*:

  delete configuration

  You have requested to Delete the system configuration
  Please confirm the request (No/Yes): **yes**

  Deleting Configuration and Rebooting
  ....
  ... se echa un rato ...
  ....
  Do you want to continue with OfficeConnect Quick Setup? 3COM−DSL>**No**
  3Com−DSL>
  Starting line test...
  3com−DSL>

  A partir de aquí *teclearemos esto*:

  add bridge network internet
  add vc internet
  set vc internet ip disable ipx disable bridging enable
  set vc internet network_service rfc_1483
  set vc internet atm vpi 8 vci 32 category_of_service unspecified pcr 0
  enable vc internet
  delete network service httpd
  delete network service tftpd
  delete network service telnetd
  save all

  Al acabar, podremos salir del minicom. Basicamente, lo que hemos hecho es activar un **bridge** con los parámetros específicos de nuestra conexión. También hemos borrado los *servicios de administración remota del router* (httpd, telnetd y tftpd) ya que no los usaremos para nada, si queremos modificar algo, usaremos de nuevo la consola serie.

  Ya tenemos nuestro router configurado, ahora hay que establecer el enlace con el equipo de linux. Esto se hace mediante ethernet, pero ojo, **no hemos asignado al router una IP** y tampoco lo haremos en el interface de linux, esto es debido a que **hemos establecido un bridge**.

Un **bridge**, muy basicamente es algo que nos permite unir transparentemente 2 redes, pasando solo el tráfico requerido entre una y otra. Entonces la idea de todo esto, es establecer un bridge entre el servidor linux y la red del proveedor ADSL a la cual accederemos mediante **PPP encapsulado bajo Ethernet** (pppoe: ppp over ethernet)

- Yo en Debian, simplemente añadí una segunda tarjeta de red, el modulo pertinente para que la cargara al arrancar y punto, **no la dí de alta** en el */etc/network/interfaces* ya que como dije, en principio **no va a llevar configuración**.
- Una vez hecho esto, lancé el script **pppoeconf** (alguna gente dice que no le rula, si es el caso, en pasos siguientes analizaremos la configuración manualmente). Este script detecta los interfaces de red instalados y uno a uno va buscando en cual hay un concentrador instalado. Si lo encuentra, nos hará unas preguntas como:
  - ♦ nombre de usuario
  - ♦ password
  - ♦ si vamos a modificar el */etc/resolv.conf* para usar los dns que nos ofrezca el proveedor.
  - ♦ si deseamos levantar la conexion al cargar linux

  Al final nos dice si queremos iniciar la conexión, le decimos que si y teoricamente, ya debemos tener acceso a internet. Podemos comprobar si es cierto visualizando el estado del interface con: **ifconfig ppp0**.

  Lo que ha hecho el pppoeconf, que tendreis que hacer **manualmente** en caso de que este no os funcione es:
  - ♦ En el archivo **/etc/ppp/pap−secrets** añadir los datos de la cuenta en una línea al final:
    *"vuestro usuario" * "vuestro password"*
  - ♦ En el archivo **/etc/ppp/peers/dsl−provider**
    user "vuestro usuario"
    pty "/usr/bin/pppoe −I ethx −T 80 −m 1452"
    (ethx es el interface del bridge mediante el cual os conectais al router)
    noipdefault
    defaultroute
  - ♦ En el archivo **/etc/ppp/ppp_on_boot.dsl**
    INTERFACE=ethx
    (ethx es el interface del bridge mediante el cual os conectais al router)
  - ♦ Si quereis levantar la conexión al arrancar, hay que crear un enlace simbólico del archivo **/etc/ppp/ppp_on_boot.dsl** que se llame **/etc/ppp/ppp_on_boot**.
  - ♦ Y bueno, nada más, podeis **conectar o desconectar** con los comandos usuales del ppp que son:
    pon dsl−provider (conectar)
    poff dsl−provider (desconectar)
- Si usais un dominio dinámico con ddclient[3], en Debian al levantar el interface de red ppp ejecuta el script que actualiza la ip del ppp con la de vuestro dominio. Sino, podeis crear uno vosotros mismos, el path es en **/etc/ppp/ip−up.d/** todo lo que metais ahi *se ejecutara al levantar la conexión*, yo personalmente he incluido ahí el scrip que me arranca el firewall bajo iptables.
- Recientemente me han comentado tambien, que los *tipos de frame ethernet de descubrimiento y sesion pppoe standard* son 8863 y 8864, sin embargo los dispositivos de 3com usan 3c12 y 3c13. Se pueden forzar manualmente en la configuracion añadiendo el parametro **"−f 3c12:3c12"** en la llamada a pppoe. A mi personalmente, me funciona sin especificarlos.

Lista de enlaces de este artículo:

1. http://packages.debian.org/ippl
2. http://packages.debian.org/ccze
3. http://packages.debian.org/ddclient
4. http://www.showmyip.com/
5. http://www.debian.org
6. http://www.knoppix.org

E−mail del autor: thesumidero@yahoo.com
Podrás encontrar este artículo e información adicional en: http://bulmalug.net/body.phtml?nIdNoticia=1758