|  |  |  |
| --- | --- | --- |
| Bochs User Manual | | |
| [Prev](http://docs.google.com/dlxlinux-networking.html) | Chapter 8. Tips and Techniques | [Next](http://docs.google.com/using-slirp.html) |

8.11. Configuring and using a tuntap network interface

If you use linux (optionally FreeBSD and Solaris, not tested), you may want to access the network through a tuntap interface. The main advantage of this interface, is that the guest has access to the host. The guest can even have access to the whole network if the host routes or masquerades the guest requests. No extra IP address is needed, all can be done using private IP addresses.

You'll find here instructions to set up Linux/Bochs to provide network access to the guest OS through a tuntap interface and private IP network. We're going to see howto :

* enable the tuntap interface in the Linux Kernel
* configure Bochs to use the tuntap interface
* set up the private network between the host and the guest
* set up the host to masquerade the guest network accesses

8.11.1. Tuntap description

From the [tuntap.txt](http://www.kernel.org/pub/linux/kernel/people/marcelo/linux-2.4/Documentation/networking/tuntap.txt) file in the Linux kernel tree :

TUN/TAP provides packet reception and transmission for user space programs.  
 It can be viewed as a simple Point-to-Point or Ethernet device, which  
 instead of receiving packets from a physical media, receives them from  
 user space program and instead of sending packets via physical media  
 writes them to the user space program.  
  
 When a program opens /dev/net/tun, driver creates and registers corresponding  
 net device tunX or tapX. After a program closed above devices, driver will  
 automatically delete tunXX or tapXX device and all routes corresponding to it.

8.11.2. Set up the linux Kernel [**[1]**](#3dy6vkm)

First make sure the tuntap module is included in the kernel :

* if you use a recent distribution, chances are that the needed modules are already build  
  Make sure that "Kernel module loader" - module auto-loading support is enabled in your kernel.  
  Add following line to the /etc/modules.conf:  
   alias char-major-10-200 tun   
    
  Run:  
   depmod -aThe driver will be automatically loaded when application access /dev/net/tun.
* Otherwise, recompile the kernel, including the configuration option  
   CONFIG\_TUN (Network device support -> Universal TUN/TAP device driver support)

**Note:** Make sure there is a /dev/net/tun device. (Can be created with '**mkdir /dev/net ; mknod /dev/net/tun c 10 200'**).

In the same way, to use masquerading, you need a kernel with the following options :

CONFIG\_IP\_NF\_CONNTRACK (Connection tracking)  
 CONFIG\_IP\_NF\_IPTABLES (IP tables support)  
 CONFIG\_IP\_NF\_NAT (Full NAT)

**Note:** Some of the other options in this group is probably also needed, (but the default setting should be OK).

8.11.3. Configure Bochs to use the tuntap interface

Make sure Bochs has one of the network adapters enabled. If you have to recompile Bochs, use --enable-ne2000 or --enable-e1000 when running **./configure** (see [Section 3.4](http://docs.google.com/compiling.html))

edit your *.bochsrc* configuration file and add something like :

ne2k: ioaddr=0x300, irq=9, mac=fe:fd:00:00:00:01,  
 ethmod=tuntap, ethdev=/dev/net/tun0, script=*/path/to/tunconfig*

Since the tuntap interface cannot be configured until a process opens it, Bochs may run a script file for you. In this case */path/to/tunconfig* should be changed to match the actual place where you'll create this script.

8.11.4. Set up the private network between the host and the guest

We'll set up a private network between the host and the guest with the following parameters:

Host IP : 192.168.1.1  
 Guest IP : 192.168.1.2

If your parameters are different, adapt the rest of the section to suit your needs.

Create the */path/to/tunconfig* script :

#!/bin/bash  
 /sbin/ifconfig ${1##/\*/} 192.168.1.1

The script get the interface name as the first parameter. Linux will forward incoming packets between interfaces.

Make it executable :

chmod 755 */path/to/tunconfig*

Run Bochs, install the guest OS, and set the following network parameters in the guest OS:

IP: 192.168.1.2  
 netmask: 255.255.255.0  
 gateway: 192.168.1.1  
 nameserver: whatever is used in linux

**Note:**  Bochs must be started by root (at least for now - the script won't have root privileges otherwise).

You may also have to edit /etc/hosts.allow in the host OS and add :

ALL: 192.168.1.2

Don't forget to set up the route on the guest.

At this point, you should be able to ping/telnet/ftp/ssh the guest from the host and vice-versa.

8.11.5. Set up the host to masquerade the guest network accesses

We are going to set up standard masquerading configuration. Edit the */path/to/tunconfig* script ans add :

/sbin/iptables -D POSTROUTING -t nat -s 192.168.1.0/24 -d ! 192.168.1.0/24 -j MASQUERADE >& /dev/null  
 /sbin/iptables -t nat -s 192.168.1.0/24 -d ! 192.168.1.0/24 -A POSTROUTING -j MASQUERADE  
 echo 1 > /proc/sys/net/ipv4/ip\_forward

**Note:**  The configuration assumes the default policy is ACCEPT (can be examined by doing '**/sbin/iptables -L**')

**Note:**  The iptables package must be installed.

And voila... The host should forward the packets of the guest to the rest of your network. You could even have access to the internet...

**Note:**  You may need to load other modules if you want to use other fancy protocols (ftp,etc...)

### Notes

|  |  |
| --- | --- |
| [[1]](http://docs.google.com/config-tuntap.html#AEN3878) | much of the information of the following section is taken from  [this email from Samuel Rydh of the Mac-On-Linux list](http://maconlinux.org/lists/mol-general/August01/0056.html) |

|  |  |  |
| --- | --- | --- |
| [Prev](http://docs.google.com/dlxlinux-networking.html) | [Home](http://docs.google.com/index.html) | [Next](http://docs.google.com/using-slirp.html) |
| Setting Up Networking in DLX Linux | [Up](http://docs.google.com/howto.html) | Using the 'slirp' networking module |