Linux IP Masquerade mini HOWTO

Ambrose Au, *ambrose@writeme.com*; David Ranch, *dranch@trinnet.net* v1.50, February 7, 1999

This document describes how to enable IP masquerade feature on a Linux host, allowing connected computers that do not have registered Internet IP addresses to connect to the Internet through your Linux box.

Contents

1 Introduction

1.1 Introduction

This document describes how to enable IP masquerade feature on a Linux host, allowing connected computers that do not have registered Internet IP addresses to connect to the Internet through your Linux box. It is possible to connect your machines to the Linux host with ethernet, as well as other kinds of connection such as a dialup ppp link. This document will emphasize on ethernet connection, since it should be the most likely case.

> **This document is intended for users using stable kernels 2.2.x and 2.0.x. Older kernels such as 1.2.x are NOT covered.**

1.2 Foreword, Feedback & Credits

I find it very confusing as a new user setting up IP masquerade on a newer kernel, i.e. 2.x kernel. Although there is a FAQ and a mailing list, there is no document dedicates on that; and there are some requests on the mailing list for such a HOWTO. So, I decided to write this up as a starting point for new users, and possibly a building block for knowledgeable users to build on for documentation. If you think I'm not doing a good job, feel free to tell me so that I can make it better.

This document is heavily based on the original FAQ by Ken Eves , and numerous helpful messages in the IP Masquerade mailing list. And a special thanks to Mr. Matthew Driver whose mailing list message inspired me to set up IP Masquerade and eventually writing this.

Please feel free to send any feedback or comments to *ambrose@writeme.com* and *dranch@trinnet.net* if we've mistaken on any information, or if any information is missing. Your invaluable feedback will certainly be influencing the future of this HOWTO!

This HOWTO is meant to be a quick guide to get your IP Masquerade working in the shortest time. As I am not a technical writer, you may find the information in this document not as general and objective as it can be. The latest news and information can be found at the *IP Masquerade Resource* <http://ipmasq.cjb.net/> web page that we maintained. If you have any technical questions on IP Masquerade, please join the IP Masquerade Mailing List instead of sending email to me since I have limited time, and the developers of IP_Masq are more capable of answering your questions.

The latest version of this document can be found at the *IP Masquerade Resource* <http://ipmasq.cjb.net/>, which also contains the HTML and postscript version:

- *http://ipmasq.cjb.net/*
- *http://ipmasq2.cjb.net/*
- Please refer to *IP Masquerade Resource Mirror Sites Listing* <http://ipmasq.cjb.net/index.html##mirror> for other mirror sites available.

1.3 Copyright & Disclaimer

The information and other contents in this document are to the best of my knowledge. However, IP Masquerade is *experimental*, and there is chance that I make mistakes as well; so you should determine if you want to follow the information in this document.

Nobody is responsible for any damage on your computers and any other losses by using the information on this document. i.e.

> **THE AUTHOR AND MAINTAINERS ARE NOT RESPONSIBLE FOR ANY DAMAGES INCURRED DUE TO ACTIONS TAKEN BASED ON THE INFORMATION IN THIS DOCUMENT.**

2 Background Knowledge

2.1 What is IP Masquerade?

IP Masquerade is a networking function in Linux. If a Linux host is connected to the Internet with IP Masquerade enabled, then computers connecting to it (either on the same LAN or connected with modems) can reach the Internet as well, even though they have *no official assigned IP addresses*.

This allows a set of machines to *invisibly* access the Internet hidden behind a gateway system, which appears to be the only system using the Internet. Breaking the security of a well set-up masquerading system should be considerably more difficult than breaking a good packet filter based firewall (assuming there are no bugs in either).

2.2 Current Status

IP Masquerade had been out for several years and is maturing as Linux heads into the 2.2.x stage. Kernels since 1.3.x had built-in support already. Many individuals and even busnesses are using it, with satisfactory results.

Browsing web pages and telnet are reported to work well over IP Masquerade. FTP, IRC and listening to Real Audio are working with certain modules loaded. Other network streaming audio such as True Speech and Internet Wave work too. Some fellow users on the mailing list even tried video conferencing software. `Ping` is now working, with the newly available ICMP patch

Please refer to section 4.3 for a more complete listing of software supported.

IP Masquerade works well with 'client machines' on several different OS and platforms. There are successful cases with systems using Unix, Windows 95, Windows NT, Windows for Workgroup(with TCP/IP package), OS/2, Macintosh System's OS with Mac TCP, Mac Open Transport, DOS with NCSA Telnet package, VAX, Alpha with Linux, and even Amiga with AmiTCP or AS225-stack. The list goes on and on, the point is, if your OS platform talks TCP/IP, it should work with IP Masquerade.

2.3 Who Can Benefit From IP Masquerade?

- If you have a Linux host connected to the Internet, and
- if you have some computers running TCP/IP connected to that Linux box on a local subnet, and/or
- if your Linux host has more than one modem and acts as a PPP or SLIP server connecting to others, which
- those **OTHER** machines do not have official assigned IP addresses. (these machines are represented by **OTHER** machines hereby)
- And of course, if you want those **OTHER** machines to make it onto the Internet without spending extra bucks :)

2.4 Who Doesn't Need IP Masquerade?

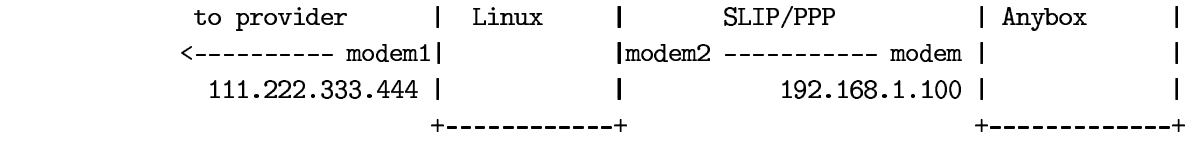
- If your machine is a stand-alone Linux host connected to the Internet, then it is pointless to have IP Masquerade running, or
- if you already have assigned addresses for your **OTHER** machines, then you don't need IP Masquerade,
- and of course, if you don't like the idea of a 'free ride'.

2.5 How IP Masquerade Works?

From IP Masquerade FAQ by Ken Eves:

```
Here is a drawing of the most simple setup:

   SLIP/PPP         +------------+                          +-------------+
```



In the above drawing a Linux box with ip_masquerading installed and running is connected to the Internet via SLIP/or/PPP using modem1. It has an assigned IP address of 111.222.333.444. It is setup that modem2 allows callers to login and start a SLIP/or/PPP connection.

The second system (which doesn't have to be running Linux) calls into the Linux box and starts a SLIP/or/PPP connection. It does NOT have an assigned IP address on the Internet so it uses 192.168.1.100. (see below)

With ip_masquerade and the routing configured properly the machine Anybox can interact with the Internet as if it was really connected (with a few exceptions).

Quoting Pauline Middelink:

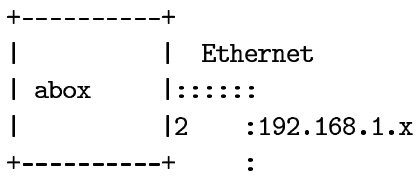
> Do not forget to mention the ANYBOX should have the Linux box as its gateway (whether is be the default route or just a subnet is no matter). If the ANYBOX can not do this, the Linux machine should do a proxy arp for all routed address, but the setup of proxy arp is beyond the scope of the document.

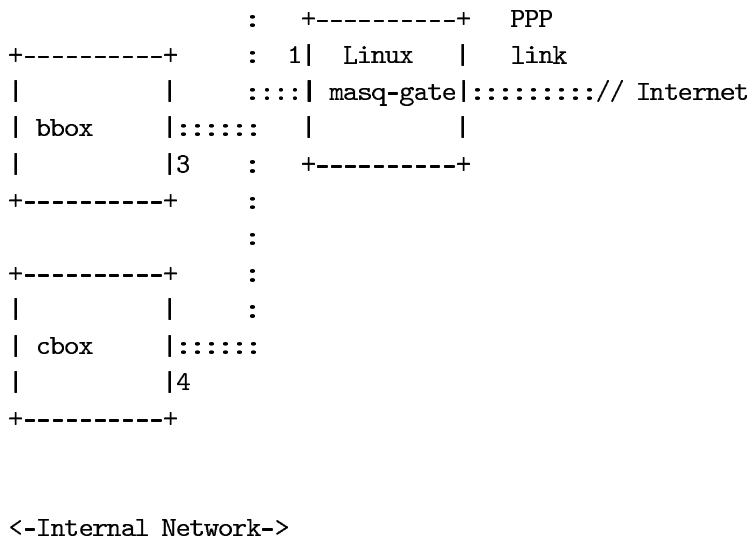
The following is an excerpt from a post on comp.os.linux.networking which has been edited to match the names used in the above example:

- I tell machine ANYBOX that my slipped linux box is its gateway.
- When a packet comes into the linux box from ANYBOX, it will assign it new source port number, and slap its own ip address in the packet header, saving the originals. It will then send the modified packet out over the SLIP/or/PPP interface to the Internet.
- When a packet comes from the Internet to the linux box, if the port number is one of those assigned above, it will get the original port and ip address, put them back in the packet header, and send the packet to ANYBOX.
- The host that sent the packet will never know the difference.

An IP Masquerading Example

typical example is given in the diagram below:-





In this example there are 4 computer systems that we are concerned about (there is presumably also something on the far right that your IP connection to the internet comes through, and there is something (far off the page) on the internet that you are interested in exchanging information with). The Linux system `masq-gate` is the masquerading gateway for the internal network of machines `abox`, `bbox` and `cbox` to get to the internet. The internal network uses one of the assigned private network addresses, in this case the class C network 192.168.1.0, with the linux box having address 192.168.1.1 and the other systems having addresses on that network.

The three machines `abox`, `bbox` and `cbox` (which can, by the way, be running any operating system as long as they can speak IP - such as **Windows 95**, **Macintosh MacTCP** or even another linux box) can connect to other machines on the internet, however the masquerading system `masq-gate` converts all of their connections so that they appear to originate from `masq-gate`, and arranges that data coming back in to a masqueraded connection is relayed back to the originating system - so the systems on the internal network see a direct route to the internet and are unaware that their data is being masqueraded.

2.6 Requirements for Using IP Masquerade on Linux 2.2.x

**** Please refer to *IP Masquerade Resource* <http://ipmasq.cjb.net/> for the latest information. ****

- Kernel 2.2.x source available from *http://www.kernel.org/*
  (Most of the modern Linux dributions such as Redhat 5.2 - shipped with 2.0.36 kernel - has modular kernel with all IP Masquerade kernel options compiled. In such cases, there is no need to compile again. If you are upgrading kernel, then you should be aware of what you need, mentioned later in the HOWTO.)

- Loadable kernel modules, preferably 2.1.121 or newer

- A well set up TCP/IP network
  covered in *Linux NET-3 HOWTO* <http://metalab.unc.edu/mdw/HOWTO/NET-3-HOWTO.html> and the *Network Administrator's Guide* <http://metalab.unc.edu/mdw/LDP/nag/nag.html>

Also check out the *Trinity OS Doc* <http://www.ecst.csuchico.edu/~dranch/LINUX/TrinityOS.wri>, a very compreshensive guide on Linux networking.

- Connectivity to Internet for your Linux host
  covered in *Linux ISP Hookup HOWTO* <http://metalab.unc.edu/mdw/HOWTO/ISP-Hookup-HOWTO.html>, *Linux PPP HOWTO* <http://metalab.unc.edu/mdw/HOWTO/PPP-HOWTO.html>, *Linux DHCP mini-HOWTO* <http://metalab.unc.edu/mdw/HOWTO/mini/DHCP.html> and *Linux Cable Modem mini-HOWTO* <http://metalab.unc.edu/mdw/HOWTO/mini/Cable-Modem.html>

- IP Chains 1.3.8 or newer available from *http://www.rustcorp.com/linux/ipchains/*
  more information on version requirement is on the *Linux IP Firewalling Chains page* <http://www.rustcorp.com/linux/ipchains/>

- For other options, please see *Linux IP Masquerade Resource* <http://ipmasq.cjb.net/>

2.7 Requirements for Using IP Masquerade on Linux 2.0.x

**** Please refer to *IP Masquerade Resource* <http://ipmasq.cjb.net/> for the latest information. ****

- Kernel 2.0.x source available from *http://www.kernel.org/*
  (Most of the modern Linux dributions such as Redhat 5.2 has modular kernel with all IP Masquerade kernel options compiled. In such cases, there is no need to compile again. If you are upgrading kernel, then you should be aware of what you need, mentioned later in the HOWTO.)

- Loadable kernel modules, preferably 2.0.0 or newer available from *http://www.pi.se/blox/modules/modules-2.0.0.tar.gz*
  (modules-1.3.57 is the minimal requirement)

- A well set up TCP/IP network
  covered in *Linux NET-3 HOWTO* <http://metalab.unc.edu/mdw/HOWTO/NET-3-HOWTO.html> and the *Network Administrator's Guide* <http://metalab.unc.edu/mdw/LDP/nag/nag.html>
  Also check out the *Trinity OS Doc* <http://www.ecst.csuchico.edu/~dranch/LINUX/TrinityOS.wri>, a very compreshensive guide on Linux networking.

- Connectivity to Internet for your Linux host
  covered in *Linux ISP Hookup HOWTO* <http://metalab.unc.edu/mdw/HOWTO/ISP-Hookup-HOWTO.html>, *Linux PPP HOWTO* <http://metalab.unc.edu/mdw/HOWTO/PPP-HOWTO.html>, *Linux DHCP mini-HOWTO* <http://metalab.unc.edu/mdw/HOWTO/mini/DHCP.html> and *Linux Cable Modem mini-HOWTO* <http://metalab.unc.edu/mdw/HOWTO/mini/Cable-Modem.html>

- Ipfwadm 2.3 or newer available from *ftp://ftp.xos.nl/pub/linux/ipfwadm/ipfwadm-2.3.tar.gz*
  more information on version requirement is on the *Linux IPFWADM page* <http://www.xos.nl/linux/ipfwadm/>

- You can optionally apply some IP Masquerade patches to enable other functionality. More information availabe on *IP Masquerade Resources* <http://ipmasq.cjb.net/> (these patches apply to all 2.0.x kernels)

3 Setting Up IP Masquerade

If your private network contains any vital information, think carefully before using IP Masquerade. This may be a GATEWAY for you to get to the Internet, and vice versa for someone on the other side of the world to get into your network.

3.1 Compiling the Kernel for IP Masquerade Support

If your Linux distribution already has the required features and modules compiled (most modular kernels will have all you need) mentioned below, then you do not have to re-compile the kernel. Reading this section is still highly recommended as it contains other useful informaiton.

3.1.1 Linux 2.2.x Kernels

- First of all, you need the kernel source for 2.2.x
- If this is your first time compiling the kernel, don't be scared. In fact, it's rather easy and it's covered in *Linux Kernel HOWTO* <http://metalab.unc.edu/mdw/HOWTO/Kernel-HOWTO.html>.
- Unpack the kernel source to `/usr/src/` with a command: `tar xvzf linux-2.2.x.tar.gz -C /usr/src`, where x is the patch level beyond 2.0
  (make sure there is a directory or symbolic link called `linux`)
- Apply appropriate patches. Since new patches are coming out, details will not be included here. Please refer to *IP Masquerade Resources* <http://ipmasq.cjb.net/> for up-to-date information.
- Refer to the Kernel HOWTO and the README file in the kernel source directory for further instructions on compiling a kernel
- Here are the options that you need to compile in: Say *YES* to the following,

```
* Prompt for development and/or incomplete code/drivers
  CONFIG_EXPERIMENTAL
  - this will allow you to select experimental IP Masquerade code compiled
    into the kernel

* Enable loadable module support
  CONFIG_MODULES
  - allows you to load ipmasq modules such as ip_masq_ftp.o

* Networking support
  CONFIG_NET

* Network firewalls
  CONFIG_FIREWALL

* TCP/IP networking
  CONFIG_INET

* IP: forwarding/gatewaying
```

```
  CONFIG_IP_FORWARD

* IP: firewalling
  CONFIG_IP_FIREWALL

* IP: masquerading
  CONFIG_IP_MASQUERADE

* IP: ipportfw masq support
  CONFIG_IP_MASQUERADE_IPPORTFW
  - recommended

* IP: ipautofw masquerade support
  CONFIG_IP_MASQUERADE_IPAUTOFW
  - optional

* IP: ICMP masquerading
  CONFIG_IP_MASQUERADE_ICMP
  - support for masquerading ICMP packets, recommended.

* IP: always defragment
  CONFIG_IP_ALWAYS_DEFRAG
  - highly recommended

* Dummy net driver support
  CONFIG_DUMMY
  - recommended

* IP: ip fwmark masq-forwarding support
  CONFIG_IP_MASQUERADE_MFW
  - optional
```

NOTE: These are just the component you need for IP Masquerade, select whatever other options you need for your specific setup.

- After compiling the kernel, you should compile and install the modules:

```
make modules; make modules_install
```

- Then you should add a few lines into your `/etc/rc.d/rc.local` file (or any file you think is appropriate) to load the required modules reside in `/lib/modules/2.2.x/ipv4/` automatically during each reboot:

```
        .
        .
        .
/sbin/depmod -a
/sbin/modprobe ip_masq_ftp
/sbin/modprobe ip_masq_raudio
/sbin/modprobe ip_masq_irc
(and other modules such as ip_masq_cuseeme, ip_masq_vdolive
 if you have applied the patches)
        .
```

.
.

IMPORTANT: IP forwarding is disabled by default in 2.2.x kernels, please make sure you enable it by running
`echo "1" > /proc/sys/net/ipv4/ip_forwarding`

For Redhat users, you may try changing `FORWARD_IPV4=false` to `FORWARD_IPV4=true` in `/etc/sysconfig/network`

- Reboot the Linux box.

3.1.2 Linux 2.0.x Kernels

- First of all, you need the kernel source (preferably the latest kernel version 2.0.36 or above)
- If this is your first time compiling the kernel, don't be scared. In fact, it's rather easy and it's covered in *Linux Kernel HOWTO* <http://metalab.unc.edu/mdw/HOWTO/Kernel-HOWTO.html>.
- Unpack the kernel source to `/usr/src/` with a command: `tar xvzf linux-2.0.x.tar.gz -C /usr/src`, where x is the patch level beyond 2.0
  (make sure there is a directory or symbolic link called `linux`)
- Apply appropriate patches. Since new patches are coming out, details will not be included here. Please refer to *IP Masquerade Resources* <http://ipmasq.cjb.net/> for up-to-date information.
- Refer to the Kernel HOWTO and the README file in the kernel source directory for further instructions on compiling a kernel
- Here are the options that you need to compile in: Say *YES* to the following,

```
* Prompt for development and/or incomplete code/drivers
  CONFIG_EXPERIMENTAL
  - this will allow you to select experimental IP Masquerade code compiled
    into the kernel

* Enable loadable module support
  CONFIG_MODULES
  - allows you to load modules

* Networking support
  CONFIG_NET

* Network firewalls
  CONFIG_FIREWALL

* TCP/IP networking
  CONFIG_INET

* IP: forwarding/gatewaying
  CONFIG_IP_FORWARD

* IP: firewalling
```

```
  CONFIG_IP_FIREWALL

* IP: masquerading (EXPERIMENTAL)
  CONFIG_IP_MASQUERADE
  - although it is experimental, it is a *MUST*

* IP: ipautofw masquerade support (EXPERIMENTAL)
  CONFIG_IP_MASQUERADE_IPAUTOFW
  -recommended

* IP: ICMP masquerading
  CONFIG_IP_MASQUERADE_ICMP
  - support for masquerading ICMP packets, optional.

* IP: always defragment
  CONFIG_IP_ALWAYS_DEFRAG
  - highly recommended

* Dummy net driver support
  CONFIG_DUMMY
  - recommended
```

NOTE: These are just the component you need for IP Masquerade, select whatever other options you need for your specific setup.

- After compiling the kernel, you should compile and install the modules:

```
make modules; make modules_install
```

- Then you should add a few lines into your `/etc/rc.d/rc.local` file (or any file you think is appropriate) to load the required modules reside in `/lib/modules/2.0.x/ipv4/` automatically during each reboot:

```
        .
        .
        .
/sbin/depmod -a
/sbin/modprobe ip_masq_ftp
/sbin/modprobe ip_masq_raudio
/sbin/modprobe ip_masq_irc
(and other modules such as ip_masq_cuseeme, ip_masq_vdolive
 if you have applied the patches)
        .
        .
        .
```

IMPORTANT: IP forwarding is disabled by default since 2.0.34 kernels, please make sure you enable it by running
`echo "1" > /proc/sys/net/ipv4/ip_forward`

For Redhat users, you may try changing `FORWARD_IPV4=false` to `FORWARD_IPV4=true` in `/etc/sysconfig/network`

- Reboot the Linux box.

3.2 Assigning Private Network IP Address

Since all **OTHER** machines do not have official assigned addressees, there must be a right way to allocate address to those machines.

From IP Masquerade FAQ:

There is an RFC (#1597, probably obsolete by now) on which IP addresses are to be used on a non-connected network. There are 3 blocks of numbers set aside specifically for this purpose. One which I use is 255 Class-C subnets at 192.168.1.n to 192.168.255.n .

```
From RCF 1597:

Section 3: Private Address Space

The Internet Assigned Numbers Authority (IANA) has reserved the
following three blocks of the IP address space for private networks:

             10.0.0.0        -   10.255.255.255
             172.16.0.0      -   172.31.255.255
             192.168.0.0     -   192.168.255.255

We will refer to the first block as "24-bit block", the second as
"20-bit block", and to the third as "16-bit" block".  Note that the
first block is nothing but a single class A network number, while the
second block is a set of 16 contiguous class B network numbers, and
third block is a set of 255 contiguous class C network numbers.
```

So, if you're using a class C network, you should name your machines as 192.168.1.1, 1.92.168.1.2, 1.92.168.1.3, ..., 192.168.1.x

192.168.1.1 is usually the gateway machine, which is your Linux host connecting to the Internet. Notice that 192.168.1.0 and 192.168.1.255 are the Network and Broadcast address respectively, which are reserved. Avoid using these addresses on your machines.

3.3 Configuring the OTHER machines

Besides setting the appropriate IP address for each machine, you should also set the appropriate gateway. In general, it is rather straight forward. You simply enter the address of your Linux host (usually 192.168.1.1) as the gateway address.

For the Domain Name Service, you can add in any DNS available. The most apparent one should be the one that your Linux is using. You can optionally add any domain search suffix as well.

After you have reconfigured those IP addresses, remember to restart the appropriate services or reboot your systems.

The following configuration instructions assume that you are using a Class C network with 192.168.1.1 as your Linux host's address. Please note that 192.168.1.0 and 192.168.1.255 are reserved.

3.3.1 Configuring Windows 95

1. If you haven't installed your network card and adapter driver, do so now.
2. Go to *'Control Panel'/'Network'*.
3. Add *'TCP/IP protocol'* if you don't already have it.
4. In *'TCP/IP properties'*, goto *'IP Address'* and set IP Address to 192.168.1.x, (1 < x < 255), and then set Subnet Mask to 255.255.255.0
5. Add 192.168.1.1 as your gateway under *'Gateway'*.
6. Under *'DNS Configuration'/'DNS Server search order'* add your the DNS that your Linux host uses (usually find in `/etc/resolv.conf`). Optionally, you can add the appropriate domain search suffix.
7. Leave all the other settings as they are unless you know what you're doing.
8. Click *'OK'* on all dialog boxes and restart system.
9. `Ping` the linux box to test the network connection: *'Start/Run'*, type: `ping 192.168.1.1`
   (This is only a LAN connection testing, you can't `ping` the outside world yet.)
10. You can optionally create a `HOSTS` file in the windows directory so that you can use hostname of the machines on your LAN. There is an example called `HOSTS.SAM` in the windows directory.

3.3.2 Configuring Windows for Workgroup 3.11

1. If you haven't installed your network card and adapter driver, do so now.
2. Install the TCP/IP 32b package if you don't have it already.
3. In *'Main'/'Windows Setup'/'Network Setup'*, click on *'Drivers'*.
4. Highlight *'Microsoft TCP/IP-32 3.11b'* in the *'Network Drivers'* section, click *'Setup'*.
5. Set IP Address to 192.168.1.x (1 < x < 255), then set Subnet Mask to 255.255.255.0 and Default Gateway to 192.168.1.1
6. Do not enable *'Automatic DHCP Configuration'* and put anything in those *'WINS Server'* input areas unless you're in a Windows NT domain and you know what you're doing.
7. Click *'DNS'*, fill in the appropriate information mentioned in STEP 6 of section 3.3.1, then click *'OK'* when you're done with it.
8. Click *'Advanced'*, check *'Enable DNS for Windows Name Resolution'* and *'Enable LMHOSTS lookup'* if you're using a look up host file, similar to the one mentioned in STEP 10 of section 3.3.1
9. Click *'OK'* on all dialog boxes and restart system.
10. `Ping` the linux box to test the network connection: *'File/Run'*, type: `ping 192.168.1.1`
    (This is only a LAN connection testing, you can't `ping` the outside world yet.)

3.3.3 Configuring Windows NT

1. If you haven't installed your network card and adapter driver, do so now.
2. Go to *'Main'/'Control Panel'/'Network'*
3. Add the TCP/IP Protocol and Related Component from the *'Add Software'* menu if you don't have TCP/IP service installed already.
4. Under *'Network Software and Adapter Cards'* section, highlight *'TCP/IP Protocol'* in the *'Installed Network Software'* selection box.
5. In *'TCP/IP Configuration'*, select the appropriate adapter, e.g. `[1]Novell NE2000 Adapter`. Then set the IP Address to 192.168.1.x ($1 < x < 255$), then set Subnet Mask to 255.255.255.0 and Default Gateway to 192.168.1.1
6. Do not enable *'Automatic DHCP Configuration'* and put anything in those *'WINS Server'* input areas unless you're in a Windows NT domain and you know what you're doing.
7. Click *'DNS'*, fill in the appropriate information mentioned in STEP 6 of section 3.3.1, then click *'OK'* when you're done with it.
8. Click *'Advanced'*, check *'Enable DNS for Windows Name Resolution'* and *'Enable LMHOSTS lookup'* if you're using a look up host file, similar to the one mentioned in STEP 10 of section 3.3.1
9. Click *'OK'* on all dialog boxes and restart system.
10. `Ping` the linux box to test the network connection: *'File/Run'*, type: `ping 192.168.1.1`
    (This is only a LAN connection testing, you can't `ping` the outside world yet.)

3.3.4 Configuring UNIX Based Systems

1. If you haven't installed your network card and recompile your kernel with the appropriate adapter driver, do so now.
2. Install TCP/IP networking, such as the nettools package, if you don't have it already.
3. Set *IPADDR* to 192.168.1.x ($1 < x < 255$), then set *NETMASK* to 255.255.255.0, *GATEWAY* to 192.168.1.1, and *BROADCAST* to 192.168.1.255
   For example, you can edit the `/etc/sysconfig/network-scripts/ifcfg-eth0` file on a Red Hat Linux system, or simply do it through the Control Panel.
   (it's different in SunOS, BSDi, Slackware Linux, etc...)
4. Add your domain name service (DNS) and domain search suffix in `/etc/resolv.conf`
5. You may want to update your `/etc/networks` file depending on your settings.
6. Restart the appropriate services, or simply restart your system.
7. Issue a `ping` command: `ping 192.168.1.1` to test the connection to your gateway machine.
   (This is only a LAN connection testing, you can't `ping` the outside world yet.)

3.3.5 Configuring DOS using NCSA Telnet package

1. If you haven't installed your network card, do so now.
2. Load the appropriate packet driver. For an NE2000 card, issue `nwpd 0x60 10 0x300`, with your network card set to IRQ 10 and hardware address at 0x300
3. Make a new directory, and then unpack the NCSA Telnet package: `pkunzip tel2308b.zip`
4. Use a text editor to open the `config.tel` file
5. Set `myip=192.168.1.x` ($1 < x < 255$), and netmask=255.255.255.0
6. In this example, you should set `hardware=packet, interrupt=10, ioaddr=60`
7. You should have at least one individual machine specification set as the gateway, i.e. the Linux host:

```
name=default
host=yourlinuxhostname
hostip=192.168.1.1
gateway=1
```

8. Have another specification for a domain name service:

```
name=dns.domain.com ; hostip=123.123.123.123; nameserver=1
```

   Note: substitute the appropriate information about the DNS that your Linux host uses

9. Save your `config.tel` file
10. Telnet to the linux box to test the network connection: `telnet 192.168.1.1`

3.3.6 Configuring MacOS Based System Running MacTCP

1. If you haven't installed the appropriate driver software for your Ethernet adapter, now would be a very good time to do so.
2. Open the *MacTCP control panel*. Select the appropriate network driver (Ethernet, NOT EtherTalk) and click on the *'More...'* button.
3. Under *'Obtain Address:'*, click *'Manually'*.
4. Under *'IP Address:'*, select *class C* from the popup menu. Ignore the rest of this section of the dialog box.
5. Fill in the appropriate information under *'Domain Name Server Information:'*.
6. Under *'Gateway Address:'*, enter 192.168.1.1
7. Click *'OK'* to save the settings. In the main window of the *MacTCP control panel*, enter the IP address of your Mac (192.168.1.x, $1 < x < 255$) in the *'IP Address:'* box.
8. Close the *MacTCP control panel*. If a dialog box pops up notifying you to do so, restart the system.

9. You may optionally ping the Linux box to test the network connection. If you have the freeware program *MacTCP Watcher*, click on the *'Ping'* button, and enter the address of your Linux box (192.168.1.1) in the dialog box that pops up. (This is only a LAN connection testing, you can't ping the outside world yet.)

10. You can optionally create a `Hosts` file in your System Folder so that you can use the hostnames of the machines on your LAN. The file should already exist in your System Folder, and should contain some (commented-out) sample entries which you can modify according to your needs.

3.3.7 Configuring MacOS Based System Running Open Transport

1. If you haven't installed the appropriate driver software for your Ethernet adapter, now would be a very good time to do so.

2. Open the *TCP/IP Control Panel* and choose *'User Mode ...'* from the *Edit* menu. Make sure the user mode is set to at least *'Advanced'* and click the *'OK'* button.

3. Choose *'Configurations...'* from the *File* menu. Select your *'Default'* configuration and click the *'Duplicate...'* button. Enter 'IP Masq' (or something to let you know that this is a special configuration) in the *'Duplicate Configuration'* dialog, it will probably say something like *'Deafault copy'*. Then click the *'OK'* button, and the *'Make Active'* button

4. Select *'Ethernet'* from the *'Connect via:'* pop-up.

5. Select the appropriate item from the *'Configure:'* pop-up. If you don't know which option to choose, you probably should re-select your *'Default'* configuration and quit. I use *'Manually'*.

6. Enter the IP address of your Mac (192.168.1.x, $1 < x < 255$) in the *'IP Address:'* box.

7. Enter 255.255.255.0 in the *'Subnet mask:'* box.

8. Enter 192.168.1.1 in the *'Router address:'* box.

9. Enter the IP addresses of your domain name servers in the *'Name server addr.:'* box.

10. Enter the name of your Internet domain (e.g. 'microsoft.com') in the *'Starting domain name'* box under *'Implicit Search Path:'*.

11. The following procedures are optional. Incorrect values may cause erratic behavior. If your not sure, it's probably better to leave them blank, unchecked and/or un- selected. Remove any information from those fields, if necessary. As far as I know there is no way through the TCP/IP dialogs, to tell the system not to use a previously select alternate "Hosts" file. If you know, I would be interested.
Check the *'802.3'* if your network requires 802.3 frame types.

12. Click the *'Options...'* button to make sure that the TCP/IP is active. I use the *'Load only when needed'* option. If you run and quit TCP/IP applications many times without rebooting your machine, you may find that unchecking the *'Load only when needed'* option will prevent/reduce the effects on your machines memory management. With the item unchecked the TCP/IP protocol stacks are always loaded and available for use. If checked, the TCP/IP stacks are automatically loaded when needed and un- loaded when not. It's the loading and unloading process that can cause your machines memory to become fragmented.

13. You may ping the Linux box to test the network connection. If you have the freeware program *MacTCP Watcher*, click on the *'Ping'* button, and enter the address of your Linux box (192.168.1.1) in the dialog box that pops up. (This is only a LAN connection testing, you can't ping the outside world yet.)

14. You can create a `Hosts` file in your System Folder so that you can use the hostnames of the machines on your LAN. The file may or may not already exist in your System Folder. If so, it should contain some (commented-out) sample entries which you can modify according to your needs. If not, you can get a copy of the file from a system running MacTCP, or just create your own (it follows a subset of the Unix `/etc/hosts` file format, described on RFC952). Once you've created the file, open the *TCP/IP control panel*, click on the *'Select Hosts File...'* button, and open the `Hosts` file.

15. Click the close box or choose *'Close'* or *'Quit'* from the *File* menu, and then click the *'Save'* button to save the changes you have made.

16. The changes take effect immediately, but rebooting the system won't hurt.

3.3.8 Configuring Novell network using DNS

1. If you haven't installed the appropriate driver software for your Ethernet adapter, now would be a very good time to do so.

2. Downloaded tcpip16.exe from `<ftp.novell.com/pub/updates/unixconn/lwp5>`

3. `edit c:\nwclient\startnet.bat`

   : (here is a copy of mine)

```
SET NWLANGUAGE=ENGLISH
LH LSL.COM
LH KTC2000.COM
LH IPXODI.COM
LH tcpip
LH VLM.EXE
F:
```

4. `edit c:\nwclient\net.cfg`

   : (change link driver to yours i.e. NE2000)

```
Link Driver KTC2000
        Protocol IPX 0 ETHERNET_802.3
        Frame ETHERNET_802.3
        Frame Ethernet_II
        FRAME Ethernet_802.2

NetWare DOS Requester
           FIRST NETWORK DRIVE = F
           USE DEFAULTS = OFF
           VLM = CONN.VLM
           VLM = IPXNCP.VLM
           VLM = TRAN.VLM
```

```
            VLM = SECURITY.VLM
            VLM = NDS.VLM
            VLM = BIND.VLM
            VLM = NWP.VLM
            VLM = FIO.VLM
            VLM = GENERAL.VLM
            VLM = REDIR.VLM
            VLM = PRINT.VLM
            VLM = NETX.VLM

    Link Support
            Buffers 8 1500
            MemPool 4096

    Protocol TCPIP
            PATH SCRIPT     C:\NET\SCRIPT
            PATH PROFILE    C:\NET\PROFILE
            PATH LWP_CFG    C:\NET\HSTACC
            PATH TCP_CFG    C:\NET\TCP
            ip_address      xxx.xxx.xxx.xxx
            ip_router       xxx.xxx.xxx.xxx
```

5. and finally created

   ```
   c:\bin\resolv.cfg

   :

   SEARCH DNS HOSTS SEQUENTIAL
   NAMESERVER 207.103.0.2
   NAMESERVER 207.103.11.9
   ```

6. I hope this helps some people get their Novell Nets online, BTW this can be done using Netware 3.1x or 4.x

3.3.9 Configuring OS/2 Warp

1. If you haven't installed the appropriate driver software for your Ethernet adapter, now would be a very good time to do so.
2. Install the TCP/IP protocoll if you don't have it already.
3. Go to *Programms/TCP/IP (LAN) / TCP/IP* Settings
4. In *'Network'* add your TCP/IP Address and set your Netmask (255.255.255.0)
5. Under *'Routing'* press *'Add'*. Set the *Type* to *'default'* and type the IP Address of your Linux Box in the Field *'Router Address'*. (192.168.1.1).
6. Set the same DNS (Nameserver) Address that your Linux host uses in *'Hosts'*.

7. Close the TCP/IP control panel. Say yes to the following question(s).
8. Reboot your system
9. You may ping the Linux box to test the network configuration. Type `'ping 192.168.1.1'` in a 'OS/2 Command prompt Window'. When ping packets are received all is ok.

3.3.10 Configuring Other Systems

The same logic should apply to setting up other platforms. Consult the sections above. If you're interested in writing about any of systems that have not been covered yet, please send a detail setup instruction to *ambrose@writeme.com* and *dranch@trinnet.net*.

3.4 Configuring IP Forwarding Policies

At this point, you should have your kernel and other required packages installed, as well as your modules loaded. Also, the IP addresses, gateway, and DNS should be all set on the **OTHER** machines.

Now, the only thing left to do is to use the IP firewalling tools to forward appropriate packets to the appropriate machine:

> ** This can be accomplished in many different ways. The following suggestions and examples worked for me, but you may have different ideas, please refer to section 4.4 and the ipchains(2.2.x) / ipfwadm(2.0.x) manpages for details. **
>
> **** This section ONLY provides you with the bare minimum rule set to get IP Masquerade working while security issue is not being considered. It is highly recomended that you spend some time to apply appropriate firewall rules to tighten security. ****

3.4.1 Linux 2.2.x Kernels

ipfwadm is no longer the tool for manipulating ipmasq rules for 2.2.x kernels, please use ipchains.

```
ipchains -P forward DENY
ipchains -A forward -s yyy.yyy.yyy.yyy/x -j MASQ
```

where x is one of the following numbers according to the class of your subnet, and yyy.yyy.yyy.yyy is your network address.

```
netmask         | x  | Subnet
~~~~~~~~~~~~~~~~|~~~~|~~~~~~~~~~~~~~~
255.0.0.0       | 8  | Class A
255.255.0.0     | 16 | Class B
255.255.255.0   | 24 | Class C
255.255.255.255 | 32 | Point-to-point
```

You may also use the format yyy.yyy.yyy.yyy/xxx.xxx.xxx.xxx, where xxx.xxx.xxx.xxx specfies your subnet mask such as 255.255.255.0

For example, if I'm on a class C subnet, I would have entered:

```
ipchains -P forward DENY
ipchains -A forward -s 192.168.1.0/24 -j MASQ

or

ipchains -P forward DENY
ipchains -A forward -s 192.168.1.0/255.255.255.0 -j MASQ
```

You can also do it on a per machine basis. For example, if I want 192.168.1.2 and 192.168.1.8 to have access to the Internet, but not the other machines, I would have entered:

```
ipchains -P forward DENY
ipchains -A forward -s 192.168.1.2/32 -j MASQ
ipchains -A forward -s 192.168.1.8/32 -j MASQ
```

Do **not** make your default policy be masquerading - otherwise someone who can manipulate their routing will be able to tunnel straight back through your gateway, using it to masquerade their identity!

Again, you can add these lines to the `/etc/rc.local` files, one of the rc files you prefer, or do it manually every time you need IP Masquerade.

For detail ipchains usage, please refer to the *Linux IPCHAINS HOWTO* <http://metalab.unc.edu/mdw/HOWTO/IPCHAINS-HOWTO.html>

3.4.2 Linux 2.0.x Kernels

```
ipfwadm -F -p deny
ipfwadm -F -a m -S yyy.yyy.yyy.yyy/x -D 0.0.0.0/0

or

ipfwadm -F -p deny
ipfwadm -F -a masquerade -S yyy.yyy.yyy.yyy/x -D 0.0.0.0/0
```

where x is one of the following numbers according to the class of your subnet, and yyy.yyy.yyy.yyy is your network address.

```
netmask         | x  | Subnet
~~~~~~~~~~~~~~~~|~~~~|~~~~~~~~~~~~~~~
255.0.0.0       | 8  | Class A
255.255.0.0     | 16 | Class B
255.255.255.0   | 24 | Class C
255.255.255.255 | 32 | Point-to-point
```

You may also use the format yyy.yyy.yyy.yyy/xxx.xxx.xxx.xxx, where xxx.xxx.xxx.xxx specfies your subnet mask such as 255.255.255.0

For example, if I'm on a class C subnet, I would have entered:

```
ipfwadm -F -p deny
ipfwadm -F -a m -S 192.168.1.0/24 -D 0.0.0.0/0
```

Since bootp request packets comes without valid IP's once the client knows nothing about it, for people with a bootp server in the masquerade/firewall machine it is necessary to use the following before the deny command:

```
ipfwadm -I -a accept -S 0/0 68 -D 0/0 67 -W bootp_clients_net_if_name -P udp
```

You can also do it on a per machine basis. For example, if I want 192.168.1.2 and 192.168.1.8 to have access to the Internet, but not the other machines, I would have entered:

```
ipfwadm -F -p deny
ipfwadm -F -a m -S 192.168.1.2/32 -D 0.0.0.0/0
ipfwadm -F -a m -S 192.168.1.8/32 -D 0.0.0.0/0
```

What appears to be a common mistake is to make the first command be this

```
ipfwadm -F -p masquerade
```

Do **not** make your default policy be masquerading - otherwise someone who can manipulate their routing will be able to tunnel straight back through your gateway, using it to masquerade their identity!

Again, you can add these lines to the `/etc/rc.local` files, one of the rc files you prefer, or do it manually every time you need IP Masquerade.

Please read section 4.4 for a detail guide on Ipfwadm

3.5 Testing IP Masquerade

It's time to give it a try, after all these hard work. Make sure the connection of your Linux hosts to the Internet is okay.

You can try browsing some *'INTERNET!!!'* web sites on your **OTHER** machines, and see if you get it. I recommend using an IP address rather than a hostname on your first try, because your DNS setup may not be correct.

For example, you can access the Linux Documentation Project site http://metalab.unc.edu/mdw/linux.html with an entry of *http://152.19.254.81/mdw/linux.html*

If you see The Linux Documentation Project homepage, then congratulations! It's working! You may then try one with hostname entry, and then ping, telnet, ssh, ftp, Real Audio, True Speech, whatever supported by IP Masquerade.....

So far, I have no trouble with the above settings, and it's full credit to the people who spend their time making this wonderful feature working.

4 Other IP Masquerade Issues and Software Support

4.1 Problems with IP Masquerade

Some protocols will not currently work with masquerading because they either assume things about port numbers, or encode data in their data stream about addresses and ports - these latter protocols need specific proxies built into the masquerading code to make them work.

4.2 Incoming services

Masquerading cannot handle incoming services at all. There are a few ways of allowing them, but they are completely separate from masquerading, and are really part of standard firewall practice.

If you do not require high levels of security then you can simply redirect ports. There are various ways of doing this - I use a modified redir program (which I hope will be available from sunsite and mirrors soon). If you wish to have some level of authorisation on incoming connections then you can either use TCP wrappers or Xinetd on top of redir (0.7 or above) to allow only specific IP addresses through, or use some other tools. The TIS Firewall Toolkit is a good place to look for tools and information.

More details can be found at *IP Masquerade Resource* <http://ipmasq.cjb.net>.

A section on more about forwarding services will be added soon.

4.3 Supported Client Software and Other Setup Note

**** The following list is not being maintained anymore. Please refer to *this page* <http://dijon.nais.com/~nevo/masq/> on applications that work thru Linux IP masquerading and *IP Masquerade Resource* <http://ipmasq.cjb.net/> for more detail. ****

Generally, application that uses TCP and UDP should work. If you have any suggestion, hints, or questions about applications with IP Masquerade, please visit this page on *applications that work thru Linux IP masquerading* <http://dijon.nais.com/~nevo/masq/> by Lee Nevo.

4.3.1 Clients that Work

General Clients

HTTP

all supported platforms, surfing the web

POP & SMTP

all supported platforms, email client

Telnet

all supported platforms, remote session

FTP

all supported platforms, with ip_masq_ftp.o module (not all sites work with certain clients; e.g. some sites cannot be reached using ws_ftp32 but works with netscape)

Archie

all supported platforms, file searching client (not all archie clients are supported)

NNTP (USENET)

all supported platforms, USENET news client

VRML

Windows(possibly all supported platforms), virtual reality surfing

traceroute

mainly UNIX based platforms, some variations may not work

ping

all platforms, with ICMP patch

anything based on IRC

all supported platforms, with ip_masq_irc.o modules

Gopher client

all supported platforms

WAIS client

all supported platforms

Multimedia Clients

Real Audio Player

Windows, network streaming audio, with ip_masq_raudio module loaded

True Speech Player 1.1b

Windows, network streaming audio

Internet Wave Player

Windows, network streaming audio

Worlds Chat 0.9a

Windows, Client-Server 3D chat program

Alpha Worlds

Windows, Client-Server 3D chat program

Internet Phone 3.2

Windows, Peer-to-peer audio communications, people can reach you only if you initiate the call, but people cannot call you

Powwow

Windows, Peer-to-peer Text audio whiteboard communications, people can reach you only if you initiate the call, but people cannot call you

CU-SeeMe

all supported platforms, with cuseeme modules loaded, please see *IP Masquerade Resource* <http://ipmasq.cjb.net/> for detail

VDOLive

Windows, with vdolive patch

Note: Some clients such as IPhone and Powwow may work even if you're not the one who initiate the call by using *ipautofw package* (refer to section 4.6)

Other Clients

NCSA Telnet 2.3.08

DOS, a suite containing telnet, ftp, ping, etc.

PC-anywhere for windows 2.0

MS-Windows, Remotely controls a PC over TCP/IP, only work if it is a client but not a host

Socket Watch

uses ntp - network time protocol

Linux net-acct package

Linux, network administration-account package

4.3.2 Clients that do not Work

Intel Internet Phone Beta 2

Connects but voice travels one way (out) Traffic only

Intel Streaming Media Viewer Beta 1

Cannot connect to server

Netscape CoolTalk

Cannot connect to opposite side

talk,ntalk

will not work - requires a kernel proxy to be written.

WebPhone

Cannot work at present (it makes invalid assumptions about addresses).

X

Untested, but I think it cannot work unless someone builds an X proxy, which is probably an external program to the masquerading code. One way of making this work is to use **ssh** as the link and use the internal X proxy of that to make things work!

4.3.3 Platforms/OS Tested as on OTHER machines

- Linux
- Solaris
- Windows 95
- Windows NT (both workstation and server)
- Windows For Workgroup 3.11 (with TCP/IP package)
- Windows 3.1 (with Chameleon package)
- Novel 4.01 Server
- OS/2 (including Warp v3)
- Macintosh OS (with MacTCP or Open Transport)
- DOS (with NCSA Telnet package, DOS Trumpet works partially)
- Amiga (with AmiTCP or AS225-stack)
- VAX Stations 3520 and 3100 with UCX (TCP/IP stack for VMS)
- Alpha/AXP with Linux/Redhat
- SCO Openserver (v3.2.4.2 and 5)
- IBM RS/6000 running AIX

Basically all OS platforms support TCP/IP and give you the option to specify the gateway/router should work with IP Masquerade.

4.4 IP Firewall Administration (ipfwadm)

This section provides a more in-depth guide on using ipfwadm.

This is a setup for a firewall/masquerade system behind a PPP link with a static PPP address follows. Trusted interface is 192.168.255.1, PPP interface has been changed to protect the guilty :). I listed each incoming and outgoing interface individually to catch IP spoofing as well as stuffed routing and/or masquerading. Also anything not explicitly allowed is forbidden!

```
#!/bin/sh
#
# /etc/rc.d/rc.firewall, define the firewall configuration, invoked from
# rc.local.
#

PATH=/sbin:/bin:/usr/sbin:/usr/bin

# testing, wait a bit then clear all firewall rules.
```

```
# uncomment following lines if you want the firewall to automatically
# disable after 10 minutes.
# (sleep 600; \
# ipfwadm -I -f; \
# ipfwadm -I -p accept; \
# ipfwadm -O -f; \
# ipfwadm -O -p accept; \
# ipfwadm -F -f; \
# ipfwadm -F -p accept; \
# ) &

# Incoming, flush and set default policy of deny. Actually the default policy
# is irrelevant because there is a catch all rule with deny and log.
ipfwadm -I -f
ipfwadm -I -p deny
# local interface, local machines, going anywhere is valid
ipfwadm -I -a accept -V 192.168.255.1 -S 192.168.0.0/16 -D 0.0.0.0/0
# remote interface, claiming to be local machines, IP spoofing, get lost
ipfwadm -I -a deny -V your.static.PPP.address -S 192.168.0.0/16 -D 0.0.0.0/0 -o
# remote interface, any source, going to permanent PPP address is valid
ipfwadm -I -a accept -V your.static.PPP.address -S 0.0.0.0/0 -D
your.static.PPP.address/32
# loopback interface is valid.
ipfwadm -I -a accept -V 127.0.0.1 -S 0.0.0.0/0 -D 0.0.0.0/0
# catch all rule, all other incoming is denied and logged. pity there is no
# log option on the policy but this does the job instead.
ipfwadm -I -a deny -S 0.0.0.0/0 -D 0.0.0.0/0 -o

# Outgoing, flush and set default policy of deny. Actually the default policy
# is irrelevant because there is a catch all rule with deny and log.
ipfwadm -O -f
ipfwadm -O -p deny
# local interface, any source going to local net is valid
ipfwadm -O -a accept -V 192.168.255.1 -S 0.0.0.0/0 -D 192.168.0.0/16
# outgoing to local net on remote interface, stuffed routing, deny
ipfwadm -O -a deny -V your.static.PPP.address -S 0.0.0.0/0 -D 192.168.0.0/16 -o
# outgoing from local net on remote interface, stuffed masquerading, deny
ipfwadm -O -a deny -V your.static.PPP.address -S 192.168.0.0/16 -D 0.0.0.0/0 -o
# outgoing from local net on remote interface, stuffed masquerading, deny
ipfwadm -O -a deny -V your.static.PPP.address -S 0.0.0.0/0 -D 192.168.0.0/16 -o
# anything else outgoing on remote interface is valid
ipfwadm -O -a accept -V your.static.PPP.address -S your.static.PPP.address/32 -D
0.0.0.0/0
# loopback interface is valid.
ipfwadm -O -a accept -V 127.0.0.1 -S 0.0.0.0/0 -D 0.0.0.0/0
# catch all rule, all other outgoing is denied and logged. pity there is no
# log option on the policy but this does the job instead.
```

```
ipfwadm -O -a deny -S 0.0.0.0/0 -D 0.0.0.0/0 -o

# Forwarding, flush and set default policy of deny. Actually the default policy
# is irrelevant because there is a catch all rule with deny and log.
ipfwadm -F -f
ipfwadm -F -p deny
# Masquerade from local net on local interface to anywhere.
ipfwadm -F -a masquerade -W ppp0 -S 192.168.0.0/16 -D 0.0.0.0/0
# catch all rule, all other forwarding is denied and logged. pity there is no
# log option on the policy but this does the job instead.
ipfwadm -F -a deny -S 0.0.0.0/0 -D 0.0.0.0/0 -o
```

You can block traffic to a particular site using the -I, -O or -F. Remember that the set of rules are scanned top to bottom and -a means "append" to the existing set of rules so any restrictions need to come before global rules. For example (and untested) :-

Using -I rules. Probably the fastest but it only stops the local machines, the firewall itself can still access the "forbidden" site. Of course you might want to allow that combination.

```
... start of -I rules ...
# reject and log local interface, local machines going to 204.50.10.13
ipfwadm -I -a reject -V 192.168.255.1 -S 192.168.0.0/16 -D 204.50.10.13/32 -o
# local interface, local machines, going anywhere is valid
ipfwadm -I -a accept -V 192.168.255.1 -S 192.168.0.0/16 -D 0.0.0.0/0
... end of -I rules ...
```

Using -O rules. Slowest because the packets go through masquerading first but this rule even stops the firewall accessing the forbidden site.

```
... start of -O rules ...
# reject and log outgoing to 204.50.10.13
ipfwadm -O -a reject -V your.static.PPP.address -S your.static.PPP.address/32 -D
204.50.10.13/32 -o
# anything else outgoing on remote interface is valid
ipfwadm -O -a accept -V your.static.PPP.address -S your.static.PPP.address/32 -D
0.0.0.0/0
... end of -O rules ...
```

Using -F rules. Probably slower than -I and this still only stops masqueraded machines (i.e. internal), firewall can still get to forbidden site.

```
... start of -F rules ...
# Reject and log from local net on PPP interface to 204.50.10.13.
ipfwadm -F -a reject -W ppp0 -S 192.168.0.0/16 -D 204.50.10.13/32 -o
# Masquerade from local net on local interface to anywhere.
ipfwadm -F -a masquerade -W ppp0 -S 192.168.0.0/16 -D 0.0.0.0/0
... end of -F rules ...
```

No need for a special rule to allow 192.168.0.0/16 to go to 204.50.11.0, it is covered by the global rules.

There is more than one way of coding the interfaces in the above rules. For example instead of -V 192.168.255.1 you can code -W eth0, instead of -V your.static.PPP.address you can use -W ppp0. Personal choice and documentation more than anything.

4.5 IP Firewalling Chains (ipchains)

This is the firewall ruleset manipulation tool primarily intended for 2.2.x kernels (there is a patch for this to work on 2.0.x).

We will update this section to give several examples on ipchains usage soon.

See the *Linux IP Firewalling Chains page* <http://www.rustcorp.com/linux/ipchains/> and *Linux IPCHAINS HOWTO* <http://metalab.unc.edu/mdw/HOWTO/IPCHAINS-HOWTO.html> for detail.

4.6 IP Masquerade and Demand-Dial-Up

1. If you would like to setup your network to automatically dial up the Internet, the *diald* demand dial-up package will be of great utility.
2. To setup the diald, please check out the *Setting Up Diald for Linux Page* <http://home.pacific.net.sg/harish/diald.config.html>
3. Once diald and IP masq have been setup, you can go to any of the client machines and initiate a web, telnet or ftp session.
4. Diald will detect the incoming request, then dial up your ISP and establish the connection.
5. There is a timeout that will occur with the first connection. This is inevitable if you are using analog modems. The time taken to establish the modem link and the PPP connections will cause your client program to timeout. This can be avoided if you are using an ISDN connection. All you need to do is to terminate the current process on the client and restart it.

4.7 IPautofw Packet Fowarder

IPautofw <ftp://ftp.netis.com/pub/members/rlynch/ipautofw.tar.gz> is a generic forwarder of TCP and UDP for Linux masquerading. Generally to utilize a package which requires UDP, a specific ip_masq module needs to be loaded; ip_masq_raudio, ip_masq_cuseeme, ... Ipautofw acts in a more generic manner, it will forward any type of traffic including those which the application specific modules will not forward. This may create a security hole if not administered correctly.

4.8 CU-SeeMe and Linux IP-Masquerade Teeny How-To

Provided by *Michael Owings* <mailto:mikey@swampgas.com>.

4.8.1 Introduction

This section will explain the necessary steps to get CU-SeeMe (both the Cornell and White Pine versions) working together with Linux's IP-Masquerade.

CU-SeeMe is a desktop video conferencing package available for both Windows and Macintosh clients. A free version is available from *Cornell University* <http://cu-seeme.cornell.edu>. A significantly enhanced commercial version can be obtained from *White Pine Software* <http://www.wpine.com>.

IP Masquerading allows one or more workstations on a LAN to "masquerade" behind a single Linux machine connected to the Internet. The workstations on the LAN can access the Internet almost transparently even without valid IP addresses. The Linux box rewrites outgoing packets from the LAN to the Internet in such a way that they they appear to originate from the Linux machine. Response packets coming back in are re-written and routed back to the correct workstations on the LAN. This arrangement allows many Internet applications to run transparently from the lan workstations. For some other applications (such as CU-SeeMe), however, the Linux masquerade code needs a little help to route packets properly. This help usually comes in the form of special kernel loadable modules. For more information on IP-Masquerading, see *The Linux IP Masquerading Website* <http://www.indyramp.com/masq/>.

4.8.2 Getting It Running

First you will need a properly configured kernel. You should have full support compiled in for both IP-Masquerading and IP AutoForwarding. IP Autoforwarding is available as a config option on kernels 2.0.30 and later – you will need to patch earlier kernels. See the *Linux IP Masquerade Resource* <http://ipmasq.cjb.net> for pointers to the IP-Autoforwarding material.

Next, you will need to get the latest version of ip_masq_cuseeme.c. The latest version is available via anon ftp from *ftp://ftp.swampgas.com/pub/cuseeme/ip_masq_cuseeme.c*. This new module will also be rolled up into the kernel 2.0.31 distribution. You should replace the version in your kernel distribution with this new version. ip_masq_cuseeme.c normally resides in net/ipv4 off of the Linux source tree. You should compile and install this module.

Now, you should set up ip autoforwarding for udp ports 7648-7649 as follows:

```
ipautofw -A -r udp 7648 7649 -c udp 7648 -u
```

OR

```
ipautofw -A -r udp 7648 7649 -h www.xxx.yyy.zzz
```

The first form will allow calls to/from the last workstation to use port 7648 (the primary cu-seeme port) . The second invocation of ipautofw will allow cu-seeme calls only to/from www.xxx.yyy.zzz. I prefer the former invocation, as it is more flexible because there is no need to specify a fixed workstation IP. However, this invocation also requires a workstation to have previously placed an outgoing call in order to receive incoming calls.

Note that both invocations leave UDP ports 7648-7649 on the client machines open to the outside world – and while this does not pose an enormous security hazard, you should use appropriate caution.

Finally, load up the new ip_masq_cuseeme module as follows:

```
modprobe ip_masq_cuseeme
```

You should now be able to fire up CU-SeeMe from a masqueraded machine on your lan and connect to a remote reflector, or another CU-Seeme user. You should also be able to get incoming calls. Note that outside callers should call using the ip of your linux gateway, NOT the masqueraded workstation.

4.8.3 Restrictions/Caveats

Password Protected Reflectors No way, no how. Uh-uh. Negatory. White Pine uses the source IP (as computed by the client program) to encrypt the password prior to transmission. Since we have to rewrite this address, the reflector ends up using the wrong source IP to decrypt it, which yields an invalid password. This will only be fixed if White Pine changes their password encryption scheme (which I have suggested), or if they would be willing to make their password encryption routines public so I could add in a fix to ip_masq_cuseeme. While chances for the latter solution are vanishingly small, I would encourage anyone reading this to contact White Pine and suggest the former approach. As the traffic on this page is relatively high, I suspect we could generate enough email to get this problem moved up on White Pine's list of priorities.

Thanx to Thomas Griwenka for bringing this to my attention.

Running a Reflector You should not attempt to run a reflector on the same machine where you have ip_masq_cuseeme and ipautoforwarding for port 7648 loaded. It simply won't work, as both setups require port 7648. Either run the reflector on another Internet-reachable host, or unload CU-SeeMe client support prior to running the reflector.

Multiple CU-SeeMe Users You cannnot have multiple simultaneous CU-SeeMe users on the LAN at this time. This is due largely to CU-SeeMe's stubborn insistence on always sending to port 7648, which can only be redirected (easily) to one LAN workstation at a time.

Using the -c (control port) invocation of ipautofw above, you can avoid to having to specify a fixed workstation address allowed to use CU-SeeMe – the first workstation to send anything out on control port 7648 will be designated to receive traffic on 7648-7649. 5 minutes or so after this workstation has been inactive on port 7648, another workstation can come along and use CU-SeeMe.

Help on Setting up CU-SeeMe Feel free to email any comments or questions to *mikey@swampgas.com*. Or if you wish, you can *call me up via CU-SeeMe* <http://www.swampgas.com/vc/vc.htm>.

4.9 Other Related Tools

We will be updating this section soon to cover more ipmasq related tools such as ipportfw and masqadmin.

5 Frequently Asked Questions

If you can think of any useful FAQ, please send it to *ambrose@writeme.com* and *dranch@trinnet.net*. Please clearly state the question and an appropriate answer. Thank you!

5.1 Does IP Masquerade work with dynamically assigned IP?

Yes, it works with dynamic IP assigned by your ISP, usually by a DHCP server. As long as you have an valid Internet IP address, it should work. Of course, static IP works too.

5.2 Can I use cable modem, DSL, satellite link, etc. to connect to the Internet and use IP Masquerade?

Sure, as long as Linux supports that network interface, it should work.

5.3 What applications are supported with IP Masquerade?

It is very difficult to keep track of a list of "working applications". However, most of the normal Internet applications are supported, such as browsing the Internet (Netscape, MSIE, etc.), ftp (such as WS_FTP), Real Audio, telnet, SSH, POP3 (incoming email - Pine, Outlook), SMTP (outgoing email), etc.

Applications involving more complicated protocols or special connection methods such as video conferencing software need special helper tools.

For more detail, please see this page about *applications that work thru Linux IP masquerading* <http://dijon.nais.com/~nevo/masq/> by Lee Nevo.

5.4 How can I get IP Masquerade running on Redhat, Debian, Slackware, etc.?

No matter what Linux distribution you got, the procedures for setting up IP Masquerade mentioned in this HOWTO should apply. Some distributions may have GUI or special configuration files that make the setup easier. We try our best to write the HOWTO as general as possible.

5.5 I've just upgraded to the 2.2.x kernels, why is IP Masquerade not working?

There are several things you should check assuming your Linux ipmasq box already have proper connection to the Internet and your LAN:

- Make sure you have the necessary features and modules are compiled and loaded. See earlier sections for detail.
- Check `/usr/src/linux/Documentation/Changes` and make sure you have the minimal requirement for the network tools installed.
- Make sure you have enabled IP forwarding. Try running `echo "1" > /proc/sys/net/ipv4/ip_forwarding`.
- You should use *ipchains* <http://www.rustcorp.com/linux/ipchains/> to manipulate ipmasq and firewalling rules.
- Go through all setup and configuration again! A lot of time it's just a typo or a stupid mistake you oversee.

5.6 I've just upgraded to the kernels 2.0.30 or later, why is IP Masquerade not working?

There are several things you should check assuming your Linux ipmasq box already have proper connection to the Internet and your LAN:

- Make sure you have the necessary features and modules are compiled and loaded. See earlier sections for detail.
- Check `/usr/src/linux/Documentation/Changes` and make sure you have the minimal requirement for the network tools installed.
- Make sure you have enabled IP forwarding. Try running `echo "1" > /proc/sys/net/ipv4/ip_forward`.
- You should use *ipfwadm* <`http://www.xos.nl/`> to manipulate ipmasq and firewalling rules. You need to patch the 2.0.x kernels to use ipchains.
- Go through all setup and configuration again! A lot of time it's just a typo or a stupid mistake you oversee.

5.7 I can't get IP Masquerade to work! What options do I have for Windows Platform?

Giving up a free, reliable, high performance solution that works on minimal hardware and pay a fortune for something that needs more hardware, lower performance and less reliable? (IMHO. And yes, I have real life experience with these ;-)

Okay, it's your call. Do a web search on MS Proxy Server, Wingate, or see www.winfiles.com. Don't tell anyone I sent you.

5.8 I've checked all my configurations, I still can't get IP Masquerade to work. What should I do?

- Stay calm. Get yourself a cup of tea and have a rest, then try the suggestions mentioned below.
- Check the *IP Masquerade Mailing List Archive* <`http://home.indyramp.com/lists/masq/`>, most likely your answer is there waiting for you.
- Post your question to the IP Masquerade Mailing List, see next the FAQ for deatil. Please only try this if you cannot find the answer from the mailing list archive.
- Post your question to related Linux networking newsgroup.
- Send email to *ambrose@writeme.com* and *dranch@trinnet.net*. You have a better chance of getting a reply if you send to both of us. David is usually pretty good on replying, and I do not want to comment on my response time.
- Check your configurations again :-)

5.9 How do I join the IP Masquerade Mailing List?

Join the Linux IP Masquerading mailing list by sending an email to *masq-subscribe@indyramp.com*.

Subject and body of the message are IGNORED. This gives you every message on the list as it comes out. You are welcome to use this form if you need it, but if you can stand the digest, please choose it instead. The digest puts less of a load on the list servers. Note that you can only post from an account/address you are subscribed from.

For more commands, email *masq-help@tori.indyramp.com*.

5.10 I want to help on IP Masquerade development. What can I do?

Join the Linux IP Masquerading DEVELOPERS list and ask the great developers there, by sending an email to *masq-dev-subscribe@tori.indyramp.com* (or for a digest format, use *masq-dev-digest-subscribe@tori.indyramp.com*).

DON'T ask non IP Masquerade development related questions there!!!!

5.11 Where can I find more information on IP Masquerade?

You can find more information on IP Masquerade at the *Linux IP Masquerade Resource* <http://ipmasq.cjb.net/> that David and I also maintained. See section 6.2 for availability.

You may also find more information at *The Semi-Original Linux IP Masquerading Web Site* <http://www.indyramp.com/masq/> maintained by Indyramp Consulting, who also provided the ipmasq mailing lists.

5.12 I want to translate this HOWTO to another language, what should I do?

Make sure the lanaguage you want to translate to is not already covered by someone else, a list of available HOWTO translations is available at the *Linux IP Masquerade Resource* <http://ipmasq.cjb.net/>.

Send an email to *ambrose@writeme.com* and I will send you the SGML source of the latest version of the HOWTO.

5.13 This HOWTO seems out of date, are you still maintaining it? Can you include more information on ...? Are there any plans for making this better?

Yes, this HOWTO is still being maintained. I'm guilty of being too busy working on two jobs and don't have much time to work on this, my apology. However, with the addition of David Ranch as the HOWTO maintainer, things should improve.

If you think of a topic that could be included in the HOWTO, please send email to me and David. It will be even better if you can provide that information. I and David will include the information into the HOWTO if it is appropriate. And many thanks for your contributions.

We have a lot of new ideas and plans for improving the HOWTO, such as case studies that will cover different network setup involving IP Masquerade, more on security, ipchains usage, ipfwadm/ipchains ruleset

examples, more FAQs, more coverage on protocol and port forwarding utilities like masqadmin, etc. If you think you can help, please do. Thanks.

5.14 I got IP Masquerade working, it" great! I want to thank you guys, what can I do?

Thank the developers and appreciate the time and effort they spent on this. Send an email to us and let us know how happy you are. Introduce other people to Linux and help them when they have problems.

6 Miscellaneous

6.1 Useful Resources

- *IP Masquerade Resource page* <http://ipmasq.cjb.net/> should have enough information for setting up IP Masquerade

- *IP masquerade mailing list archive* <http://www.indyramp.com/masq/list/> contains some of the recent messages sent to the mailing list.
- This *Linux IP Masquerade mini HOWTO* <http://ipmasq.cjb.net/ipmasq-HOWTO.html> for kernel 2.2.x and 2.0.x
- *IP Masquerade HOWTO for kernel 1.2.x* <http://ipmasq.cjb.net/ipmasq-HOWTO-1.2.x.txt> if you're using an older kernel
- *IP masquerade FAQ* <http://www.indyramp.com/masq/ip_masquerade.txt> has some general information
- *Linux IPCHAINS HOWTO* <http://metalab.unc.edu/mdw/HOWTO/IPCHAINS-HOWTO.html> and *http://www.rustcorp.com/linux/ipchains/* has lots of information for ipchains usage, as well as source and binaries for the ipchains.
- *X/OS Ipfwadm page* <http://www.xos.nl/linux/ipfwadm/> contains sources, binaries, documentation, and other information about the `ipfwadm` package
- A page on *applications that work thru Linux IP masquerading* <http://dijon.nais.com/~nevo/masq/> by Lee Nevo provides tips and tricks on getting applications to work with IP Masquerade.
- The *LDP Network Administrator's Guide* <http://metalab.unc.edu/mdw/LDP/nag/nag.html> is a must for beginners trying to set up a network.
- *Trinity OS Doc* <http://www.ecst.csuchico.edu/~dranch/LINUX/TrinityOS.wri>, a very compreshensive guide on Linux networking.
- *Linux NET-3 HOWTO* <http://metalab.unc.edu/mdw/HOWTO/NET-3-HOWTO.html> also has lots of useful information about Linux networking
- *Linux ISP Hookup HOWTO* <http://metalab.unc.edu/mdw/HOWTO/ISP-Hookup-HOWTO.html> and *Linux PPP HOWTO* <http://metalab.unc.edu/mdw/HOWTO/PPP-HOWTO.html> gives you information on how to connect your Linux host to the Internet

- *Linux Ethernet-Howto* <http://metalab.unc.edu/mdw/HOWTO/Ethernet-HOWTO.html> is a good source of information about setting up a LAN running ethernet
- You may also be interested in *Linux Firewalling and Proxy Server HOWTO* <http://metalab.unc.edu/mdw/HOWTO/Firewall-HOWTO.html>
- *Linux Kernel HOWTO* <http://metalab.unc.edu/mdw/HOWTO/Kernel-HOWTO.html> will guide you through the kernel compilation process
- Other *Linux HOWTOs* <http://metalab.unc.edu/mdw/HOWTO/HOWTO-INDEX-3.html> such as Kernel HOWTO
- Posting to the USENET newsgroup: *comp.os.linux.networking*

6.2 Linux IP Masquerade Resource

The *Linux IP Masquerade Resource* <http://ipmasq.cjb.net/> is a website dedicated to Linux IP Masquerade information also maintained by David Ranch and I. It usually has the latest information related to IP Masquerade and may have information that is not being included in the HOWTO.

You may find the Linux IP Masquerade Resource at the following locations:

- *http://ipmasq.cjb.net/*, Primary Site, redirected to *http://www.tor.shaw.wave.ca/~ambrose/*
- *http://ipmasq2.cjb.net/*, Secondary Site, redirected to *http://www.geocities.com/SiliconValley/Heights/2288/*

6.3 Thanks to

- David Ranch, dranch@trinnet.net
  help maintaining this HOWTO and the Linux IP Masquerade Resource Page, ..., too many to list here :-)
- Michael Owings, mikey@swampgas.com
  on providing section for CU-SeeMe and Linux IP-Masquerade Teeny How-To
- Gabriel Beitler, gbeitler@aciscorp.com
  on providing section 3.3.8 (setting up Novel)
- Ed Doolittle, dolittle@math.toronto.edu
  on suggestion to `-V` option in `ipfwadm` command for improved security
- Matthew Driver, mdriver@cfmeu.asn.au
  on helping extensively on this HOWTO, and providing section 3.3.1 (setting up Windows 95)
- Ken Eves, ken@eves.com
  on the FAQ that provides invaluable information for this HOWTO
- Ed. Lott, edlott@neosoft.com
  for a long list of tested system and software

- Nigel Metheringham, Nigel.Metheringham@theplanet.net
  on contributing his version of IP Packet Filtering and IP Masquerading HOWTO, which make this HOWTO a better and technical in-depth document
  section 4.1, 4.2, and others
- Keith Owens, kaos@ocs.com.au
  on providing an excellent guide on ipfwadm section 4.2
  on correction to `ipfwadm -deny` option which avoids a security hole, and clarified the status of `ping` over IP Masquerade
- Rob Pelkey, rpelkey@abacus.bates.edu
  on providing section 3.3.6 and 3.3.7 (setting up MacTCP and Open Transport)
- Harish Pillay, h.pillay@ieee.org
  on providing section 4.5 (dial-on-demand using diald)
- Mark Purcell, purcell@rmcs.cranfield.ac.uk
  on providing section 4.6 (IPautofw)
- Ueli Rutishauser, rutish@ibm.net
  on providing section 3.3.9 (setting up OS/2 Warp)
- John B. (Brent) Williams, forerunner@mercury.net
  on providing section 3.3.7 (setting up Open Transport)
- Enrique Pessoa Xavier, enrique@labma.ufrj.br
  on the bootp setup suggestion
- developers of IP Masquerade for this great feature
  - Delian Delchev, delian@wfpa.acad.bg
  - Nigel Metheringham, Nigel.Metheringham@theplanet.net
  - Keith Owens, kaos@ocs.com.au
  - Jeanette Pauline Middelink, middelin@polyware.iaf.nl
  - David A. Ranch, trinity@value.net
  - Miquel van Smoorenburg, miquels@q.cistron.nl
  - Jos Vos, jos@xos.nl
  - Paul Russell, Paul.Russell@rustcorp.com.au
  - And more who I may have failed to mention here (please let me know)
- all users sending feedback and suggestion to the mailing list, especially the ones who reported errors in the document and the clients that are supported and not supported
- I appologize if I have not included information that some fellow users sent me. There are many suggestions and ideas sent to me, but I just do not have enough time to verify or I lost track of them. I am trying my best to incorporate all the information sent to me into the HOWTO. I thank you for the effort, and I hope you understand my situation.

6.4 Reference

- IP masquerade FAQ by Ken Eves
- IP masquerade mailing list archive by Indyramp Consulting
- Ipfwadm page by X/OS
- Various networking related Linux HOWTOs