Masquerading Made Simple HOWTO

John Tapsell

tapselj0@cs.man.ac.uk

Thomas Spellman

thomas@resonance.org

Matthias Grimm

DeadBull@gmx.net

Revision History

Revision 0.07	2002–02–27	Revised by: jpt
Revision 0.06	2001–09–08	Revised by: jpt
Revision 0.05	2001–09–07	Revised by: jpt
Revision 0.04	2001–09–01	Revised by: jpt
Revision 0.03	2001–07–06	Revised by: jpt

All of the authors are available on #debian on irc.opensource.net

John Tapsell (JohnFlux) is the official maintainer.

Email me (John Tapsell) for any query, flame, feedback, a date, etc.

Shamelessly stealing from David Ranch's work – `<dranch@trinnet.net>`.

This is NOT a replacement for the IP–Masquerading HOWTO – it is to complement it, and the two should be read side by side. I do not include things in here that are covered by the the other HOWTO, nor do I explain what it all means, or what it is all about. See http://ipmasq.cjb.net and the standard Masq–HOWTO for a much better guides.

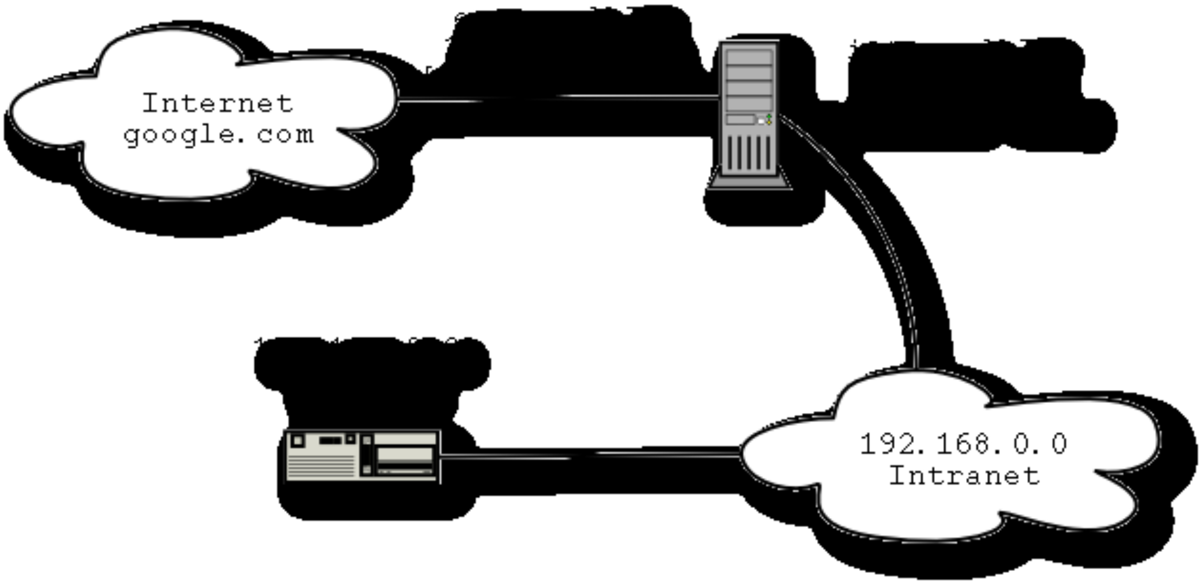
This document describes how to enable the Linux IP Masquerade feature on a given Linux host. IP Masq is a form of Network Address Translation or NAT that allows internally networked computers that do not have one or more registered Internet IP addresses to have the ability to communicate to the Internet via your Linux boxes single Internet IP address.

Table of Contents

1. Introduction

This is intentionally short and to the point.

If you have a network, that you want to attach to the outside:



2. Summary: (I like doing summaries first)

Assuming external internet card is eth0, and external IP is 123.12.23.43 and the internal network card is eth1, then:

```
$> modprobe ipt_MASQUERADE # If this fails, try continuing anyway
$> iptables -F; iptables -t nat -F; iptables -t mangle -F
$> iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 123.12.23.43
$> echo 1 > /proc/sys/net/ipv4/ip_forward
```

Or for a dial–up connection:

```
$> modprobe ipt_MASQUERADE # If this fails, try continuing anyway
$> iptables -F; iptables -t nat -F; iptables -t mangle -F
$> iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
$> echo 1 > /proc/sys/net/ipv4/ip_forward
```

Then to secure it:

```
$> iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$> iptables -A INPUT -m state --state NEW -i ! eth0 -j ACCEPT
$> iptables -P INPUT DROP   #only if the first two are succesful
$> iptables -A FORWARD -i eth0 -o eth0 -j REJECT
```

Or for a dial–up connection (with eth0 as the internal network card):

```
$> iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$> iptables -A INPUT -m state --state NEW -i ! ppp0 -j ACCEPT
$> iptables -P INPUT DROP   #only if the first two are succesful
$> iptables -A FORWARD -i ppp0 -o ppp0 -j REJECT
```

And thats it! To view the rules do "**iptables –t nat –L**"

3. Bitmore indepth version

Compiling the kernel: (Use a 2.4.x kernel or greater)

You need the following support in the kernel:

- Under Networking Options
    - Network packet filtering (CONFIG_NETFILTER)
- Under Networking Options−>Netfilter Configuration
    - Connection tracking (CONFIG_IP_NF_CONNTRACK)
    - FTP Protocol support (CONFIG_IP_NF_FTP)
    - IP tables support (CONFIG_IP_NF_IPTABLES)
    - Connection state match support (CONFIG_IP_NF_MATCH_STATE)
    - Packet filtering (CONFIG_IP_NF_FILTER)
        - REJECT target support (CONFIG_IP_NF_TARGET_REJECT)
    - Full NAT (CONFIG_IP_NF_NAT)
        - MASQUERADE target support (CONFIG_IP_NF_TARGET_MASQUERADE)
        - REDIRECT target support (CONFIG_IP_NF_TARGET_REDIRECT)
    - Packet mangling (CONFIG_IP_NF_MANGLE)
    - LOG target support (CONFIG_IP_NF_TARGET_LOG)

First, if the iptable and masq modules are not compiled into the kernel and not installed, but do exist as modules, we need to install them. If you insmod ipt_MASQUERADE it will load ip_tables, ip_conntrack and iptable_nat.

```
$> modprobe ipt_MASQERADE
```

Now either your Intranet is large, or you're just trying to get two or three machines to work on the internet – it doesn't make much difference either way.

Okay, I'm assuming that you have no other rules, so do:

```
$> iptables −F; iptables −t nat −F; iptables −t mangle −F
```

If you get an error saying can't find iptables, go find it and install it. If it says no such table 'nat', recompile the kernel with nat support. If it says no such table as 'mangle', don't worry about it, it's not necessary for MASQ'ing. If it says iptables is incompatible with your kernel, go get > 2.4 and compile that with iptables support.

Then if you have a static ip do (e.g. network card not using DHCP):

```
$> iptables −t nat −A POSTROUTING −o eth0 −j SNAT −−to 123.12.23.43
```

or for dynamic (e.g. a modem − you have to call a number first):

```
$> iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

Then finally to tell the kernel yes, you really do want to start forwarding packets: (This only needs to be done once per reboot – but dosen't hurt to do it lots)

```
$> echo 1 > /proc/sys/net/ipv4/ip_forward
```

Once you have checked this all works (See under Post–install) only allow masquerading from the internal network – you don't want to allow people on the internet to use it after all :)

First, allow any existing connections, or anything related (e.g. ftp server connecting back to you)

```
$> iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

If this gives an error, then you most likely don't have state tracking in the kernel – go recompile. Then allow new connections only from our intranet (local/internal network). Replace the ppp0 with eth0 or whatever your *external* device is. (The ! means anything but)

```
$> iptables -A INPUT -m state --state NEW -i ! ppp0 -j ACCEPT
```

And now deny everything else:

```
$> iptables -P INPUT DROP   #only if the first two are succesful
```

If either of the first two rules failed, then this last rule with prevent the masquerading from working at all. To undo this rule do "**iptables –P INPUT ACCEPT**".

4. Post−install Instructions

And it should all work now. Don't forget to:

- Setup all the clients on the internal network to point to the Linux internal IP address as their gateway. (In windows right−click network neighbourhood−>properties−>gateway then change it to the Linux gateway internal ip.)
- Setup all the clients to use your ISP's HTTP proxy if they have one, use a transparent proxy (WARNING − I've heard reports of transparent proxying to be very slow on very big networks), or run squid on your new linux gateway. (This is optional, but preferrable for large networks)
- Now you should start securing it! First turn off forwarding in general: "**iptables −P FORWARD DROP**", and then learn how to use iptables and `/etc/hosts.allow` and `/etc/hosts.deny` to secure your system. WARNING − Don't try this mentioned iptables rule until you have the masquerading working. You have to explicitely allow every packet through that you want if you are going to set the last rule to be DENY. (Undo with "**iptables −P FORWARD ACCEPT**")
- Allow through any services you do want the internet to see.

  For an example, to allow access to your web server do:

  ```
  $> iptables −A INPUT --protocol tcp --dport 80 −j ACCEPT
  $> iptables −A INPUT --protocol tcp --dport 443 −j ACCEPT
  ```

  To allow ident (For connecting to irc etc) do

  ```
  $> iptables −A INPUT --protocol tcp --dport 113 −j ACCEPT
  ```

To test it:

- Try connecting from a client to the web using an IP. Google's IP is 216.239.33.100 (well that's one of them) and you should be able to get a reply from that. e.g. "**ping 216.239.33.100**" "**lynx 216.239.33.100**".
- Try a full out connection by name. e.g. "**ping google.com**" "**lynx google.com**" or from Internet Explorer / netscape.

Where eth0 is the external Internet card, and 123.12.23.43 is the external ip of that machine.

5. FAQ's – Frequently Asked Compla^H^H^H^H^H^H Questions

- How do I list the rules I've got so far?

  – Try

```
$> iptables -L
$> iptables -t nat -L
```

- It don't work! It doesn't like iptables / NAT / SNAT / MASQ

  – Go get the latest kernel, and compile with iptables and full NAT support.

- It don't work! The masquerading doesn't work at all! Die scum!

  – Try **echo 1 > /proc/sys/net/ipv4/ip_forward**

- It don't work! I can't use the network at all and I hate you!

  – Try

```
$> iptables -F
$> iptables -t nat -F
$> iptables -t mangle -F
```

  (all rules went bye–bye) then rerun the other iptables rules.

  – Try **iptables –P FORWARD ACCEPT**

- It still don't work!

  – Hmm, does "**dmesg | tail**" give any errors? or "**cat /var/log/messages | tail**" ? Like I care tho...

- I don't get, it just ain't working!

  – I dunno.. but you should be able to:

```
        1) From the gateway machine, ping the outside
        2) From the gateway ping your internal machines
        3) From the internal machines ping the gateway
```

  And this is *before* you play with masq'ing

- Where do I put this stuff?

  – In the `/etc/network/interfaces` file, or firewall.rc. If you put it in the interfaces file, then put it as a pre–up to the external interface, and have "**iptables –t nat –F**" as the post–down.

- How do I get it to only bring the ppp up on demand?

– Assuming your ISP gateway IP is say 23.43.12.43 for arguments sake, then append a line like this:

:23.43.12.43

to `/etc/ppp/peers/provider` at the end. (this is for dynamic IP – static IP would be my.**external.ip.number:23.43.12.43**)

Then at the end of that file add on a newline:

demand

Pppd will remain in the background to redial the connection on demand if it's dropped until you do an "**ifdown ppp0**" or a "**poff**", unless you add a "**nopersist**" option, in which case pppd will exit after the connection is up. You can also add on a new line "**idle 600**" to disconnect after 10 mins of idleness.

- The connection keeps dropping!

  – First, do you have demand dialing? Is it just doing what it is supposed to? Check `/etc/ppp/peers/provider`, and make sure your dial up works fine before attempting masq'ing.

  – Secondly, if not, then perhaps, like me, something is going weird, and you need to fall back to Linux 2.4.3 and see if that works instead.. dunno why.

- I hate doing this myself! I want a pre–made script and GUI and stuff.

  – Sure: http://shorewall.sourceforge.net/

  Eat your heart out!

- Do I count Cable modems as static or dynamic IP's?

  – Good question.. might as well make it dynamic.

- Do I count DHCP network cards as static or dynamic IP's?

  – They are dynamic.

- How do I handle incomming services?

  – Try forwarding or redirecting the IP ports – again make sure you firewall this if needed.

- From the clients, I can ping the linux gateway's external IP address, but can't access the internet.

  – Okay, try doing "**rmmod iptable_filter**" – more info on this as I get it.

  – Make sure your not running *routed* or *gated* – to check run "**ps aux | grep –e routed –e gated**".

  – Look at http://ipmasq.cjb.net

- How can I view the connections establish? Something like netstat..
  - Try cat /proc/net/ip_conntrack
- I need more squid info and routing and stuff!
  - Try the Advanced Routing HOWTO http://www.linuxdoc.org/HOWTO/Adv−Routing−HOWTO.html
- This howto is crap! How do I yell at the guys who wrote this?
  - Go to #debian on irc.opensource.net and find and locate JohnFlux.
- This howto is crap! How can I see better versions?
  - Try http://ipmasq.cjb.net
  - Consult the LDP Masq−HOWTO.