Setting up IP Aliasing on A Linux Machine Mini−HOWTO

Harish Pillay

h.pillay@ieee.org

Joy Yokley − Converted document from HTML to DocBook v4.1 (SGML)

2001−01−23

Revision History

Revision 1.2	2001−01−26	Revised by: JEY
Revision 1.1	2001−01−24	Revised by: JEY
Revision 1.0	1997−01−13	Revised by: HP

This is a cookbook recipe on how to set up and run IP aliasing on a Linux box and how to set up the machine to receive e−mail on the aliased IP addresses.

Table of Contents

1. My Setup

- IP Alias is standard in kernels 2.0.x and 2.2.x, and available as a compile−time option in 2.4.x (IP Alias has been deprecated in 2.4.x and replaced by a more powerful firewalling mechanism.)
- IP Alias compiled as a loadable module. You would have indicated in the "make config" command to make your kernel, that you want the IP Masq to be compiled as a (M)odule. Check the Modules HOW−TO (if that exists) or check the info in `/usr/src/linux/Documentation/modules.txt.`
- I have to support 2 additional IPs over and above the IP already allocated to me.
- A D−Link DE620 pocket adapter (not important, works with any Linux supported network adapter).

2. Commands

1. Load the IP Alias module (you can skip this step if you compiled the module into the kernel):
   ```
   /sbin/insmod /lib/modules/`uname −r`/ipv4/ip_alias.o
   ```
2. Setup the loopback, eth0, and all the IP addresses beginning with the main IP address for the eth0 interface:
   ```
   /sbin/ifconfig lo 127.0.0.1
   /sbin/ifconfig eth0 up
   /sbin/ifconfig eth0 172.16.3.1
   /sbin/ifconfig eth0:0 172.16.3.10
   /sbin/ifconfig eth0:1 172.16.3.100
   ```

   172.16.3.1 is the main IP address, while .10 and .100 are the aliases. The magic is the eth0:x where x=0,1,2,...n for the different IP addresses. The main IP address does not need to be aliased.

3. Setup the routes. First route the loopback, then the net, and finally, the various IP addresses starting with the default (originally allocated) one:
   ```
   /sbin/route add −net 127.0.0.0
   /sbin/route add −net 172.16.3.0 dev eth0
   /sbin/route add −host 172.16.3.1 dev eth0
   /sbin/route add −host 172.16.3.10 dev eth0:0
   /sbin/route add −host 172.16.3.100 dev eth0:1
   /sbin/route add default gw 172.16.3.200
   ```

   That's it.

In the example IP address above, I am using the Private IP addresses (RFC 1918) for illustrative purposes. Substitute them with your own official or private IP addresses.

The example shows only 3 IP addresses. The max is defined to be 256 in `/usr/include/linux/net_alias.h.` 256 IP addresses on ONE card is a lot :−)!

Here's what my `/sbin/ifconfig` looks like:

```
lo        Link encap:Local Loopback
               inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
                         UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
                    RX packets:5088 errors:0 dropped:0 overruns:0
                    TX packets:5088 errors:0 dropped:0 overruns:0

eth0      Link encap:10Mbps Ethernet  HWaddr 00:8E:B8:83:19:20
                    inet addr:172.16.3.1  Bcast:172.16.3.255  Mask:255.255.255.0
                    UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
                    RX packets:334036 errors:0 dropped:0 overruns:0
                    TX packets:11605 errors:0 dropped:0 overruns:0
                    Interrupt:7 Base address:0x378

eth0:0    Link encap:10Mbps Ethernet  HWaddr 00:8E:B8:83:19:20
                    inet addr:172.16.3.10  Bcast:172.16.3.255  Mask:255.255.255.0
                    UP BROADCAST RUNNING  MTU:1500  Metric:1
                    RX packets:0 errors:0 dropped:0 overruns:0
                    TX packets:0 errors:0 dropped:0 overruns:0

eth0:1    Link encap:10Mbps Ethernet  HWaddr 00:8E:B8:83:19:20
                    inet addr:172.16.3.100  Bcast:172.16.3.255  Mask:255.255.255.0
                    UP BROADCAST RUNNING  MTU:1500  Metric:1
```

```
                    RX packets:1 errors:0 dropped:0 overruns:0
                    TX packets:0 errors:0 dropped:0 overruns:0
```

And `/proc/net/aliases`:

```
device         family    address
eth0:0           2      172.16.3.10
eth0:1           2      172.16.3.100
```

And `/proc/net/alias_types`:

```
type    name            n_attach
2       ip              2
```

Of course, the stuff in `/proc/net` was created by the **ifconfig** command and not by hand!

3. Troubleshooting: Questions and Answers

3.1. Question: How can I keep the settings through a reboot?

Answer: Whether you are using BSD−style or SysV−style (Redhat" for example) **init**, you can always include it in `/etc/rc.d/rc.local`. Here's what I have on my SysV init system (Redhat" 3.0.3 and 4.0):

My `/etc/rc.d/rc.local`: (edited to show the relevant portions)

```
#setting up IP alias interfaces
echo "Setting 172.16.3.1, 172.16.3.10, 172.16.3.100 IP Aliases ..."
/sbin/ifconfig lo 127.0.0.1
/sbin/ifconfig eth0 up
/sbin/ifconfig eth0 172.16.3.1
/sbin/ifconfig eth0:0 172.16.3.10
/sbin/ifconfig eth0:1 172.16.3.100
#setting up the routes
echo "Setting IP routes ..."
/sbin/route add −net 127.0.0.0
/sbin/route add −net 172.16.3.0 dev eth0
/sbin/route add −host 172.16.3.1 eth0
/sbin/route add −host 172.16.3.10 eth0:0
/sbin/route add −host 172.16.3.100 eth0:1
/sbin/route add default gw 172.16.3.200
#
```

3.2. Question: How do I set up the IP aliased machine to receive e−mail on the various aliased IP addresses (on a machine using sendmail)?

Answer: Create (if it doesn't already exist) a file called, `/etc/mynames.cw`,for example. The file does not have to be this exact name nor in the `/etc directory`.

In that file, place the official domain names of the aliased IP addresses. If these aliased IP addresses do not have a domain name, then you can place the IP address itself.

The `/etc/mynames.cw` might look like this:

```
# /etc/mynames.cw − include all aliases for your machine here; # is a comment
domain.one.net
domain.two.com
domain.three.org
4.5.6.7
```

In your `sendmail.cf` file, where it defines a file class macro Fw, add the following:

```
##################
#   local info   #
##################
```

```
# file containing names of hosts for which we receive email
Fw/etc/mynames.cw
```

That should do it. Test out the new setting by invoking sendmail in test mode. The following is an example:

```
ganymede$ /usr/lib/sendmail −bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter < ruleset> < address>
> 0 me@4.5.6.7
rewrite: ruleset  0   input: me @ 4 . 5 . 6 . 7
rewrite: ruleset 98   input: me @ 4 . 5 . 6 . 7
rewrite: ruleset 98 returns: me @ 4 . 5 . 6 . 7
rewrite: ruleset 97   input: me @ 4 . 5 . 6 . 7
rewrite: ruleset  3   input: me @ 4 . 5 . 6 . 7
rewrite: ruleset 96   input: me < @ 4 . 5 . 6 . 7 >
rewrite: ruleset 96 returns: me < @ 4 . 5 . 6 . 7 . >
rewrite: ruleset  3 returns: me < @ 4 . 5 . 6 . 7 . >
rewrite: ruleset  0   input: me < @ 4 . 5 . 6 . 7 . >
rewrite: ruleset 98   input: me < @ 4 . 5 . 6 . 7 . >
rewrite: ruleset 98 returns: me < @ 4 . 5 . 6 . 7 . >
rewrite: ruleset  0 returns: $# local $: me
rewrite: ruleset 97 returns: $# local $: me
rewrite: ruleset  0 returns: $# local $: me
> 0 me@4.5.6.8
rewrite: ruleset  0   input: me @ 4 . 5 . 6 . 8
rewrite: ruleset 98   input: me @ 4 . 5 . 6 . 8
rewrite: ruleset 98 returns: me @ 4 . 5 . 6 . 8
rewrite: ruleset 97   input: me @ 4 . 5 . 6 . 8
rewrite: ruleset  3   input: me @ 4 . 5 . 6 . 8
rewrite: ruleset 96   input: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset 96 returns: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset  3 returns: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset  0   input: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset 98   input: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset 98 returns: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset 95   input: < > me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset 95 returns: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset  0 returns: $# smtp $@ 4 . 5 . 6 . 8 $: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset 97 returns: $# smtp $@ 4 . 5 . 6 . 8 $: me < @ 4 . 5 . 6 . 8 >
rewrite: ruleset  0 returns: $# smtp $@ 4 . 5 . 6 . 8 $: me < @ 4 . 5 . 6 . 8 >
>
```

Notice when I tested me@4.5.6.7, it delivered the mail to the local machine, while me@4.5.6.8 was handed off to the smtp mailer. That is the correct response.

You are all set now.

4. Acknowledgements

Thanks to all those who have done this great work on Linux and IP Aliasing. And especially to Juan Jose Ciarlante for clarifying my questions.

Kudos to the ace programmers!

If you find this document useful or have suggestions on improvements, email me at <`h.pillay@ieee.org`>.

Enjoy.

For additional information on networking, you may want to consult the The Linux Networking Overview HOWTO.