Perth Linux Users Group -- December 2003

Tony Breeds <magni@plug.linux.org.au>

Presents

Introduction to IP[v4] networking Part 3, routing

TOC

Apologies

Revision from part 1

Why

IP Addressing

Niceties

- DHCP
- DNS

So I assume your network at home is:

- operating on 192.168.1.1/24
- has one linux box (the "server") [.1]
- running DNS and DHCP services.
- and one other machine, either Linux or Windows [.16]

Revision from part 2

Samba

ssh
Webserver (apache)

Quake I/II/III
Neverwinter Nights

ISP Hookup

Again all Distro's have a different way of doing this.

Assuming a std. dialup account

You need

- Phone number
- Username
- Password
- install pppd (usually in the "ppp" package)

ISP Hookup - chatscript

Create/edit /etc/ppp/peers/ISP-NAME.chat

```
ABORT BUSY
ABORT 'NO CARRIER'
ABORT VOICE
ABORT 'NO DIALTONE'
ABORT 'NO DIAL TONE'
ABORT 'NO ANSWER'
ABORT DELAYED
'' ATZm0l0
OK ATDT PHONE-NUMBER
CONNECT \d\c
```

ISP Hookup - peers

Create/edit /etc/ppp/peers/ISP-NAME

```
hide-password
noauth
connect "/usr/sbin/chat -v -f /etc/ppp/peers/ISP-NAME.chat"
/dev/ttyS0
115200
defaultroute
user YOUR-USERNAME-HERE
remotename ISP-NAME
holdoff 15
```

ISP Hookup

Create/edit /etc/ppp/pap-secrets

```
"YOUR-USERNAME"      "ISP-NAME"     "PASSWORD"
```

Now connect up (as root)

```
# pppd call ISP-NAME
```

Assuming all went well you should now be connected to the 'Net

Firewalling - 1

At this point ANYONE on the 'Net can connect to all your cool network services.

- You probably want a firewall.
- In kernel 2.4 and 2.6 the firewalling tool is "iptables"

```
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Firewalling - 1

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Firewalling - basics

Basic commands for iptables

- iptables -F
- iptables -X
- iptables -N NAME
- iptables -L
- iptables {-A,-I,-D}

Predefined tables

- ACCEPT
- DROP
- REJECT

Firewalling - Getting started

You might be tempted to jump in and do:

```
iptables -A INPUT -j ACCEPT -p tcp --dport 22
iptables -A INPUT -j DROP
```

but that would be bad.

```
iptables -A INPUT -j ACCEPT -i lo
iptables -A INPUT -j ACCEPT -i eth0
iptables -A INPUT -j ACCEPT -i ppp0 -p tcp --dport 22
iptables -A INPUT -j DROP
```

Would be better

```
iptables -A INPUT -j ACCEPT -i lo
iptables -A INPUT -j ACCEPT -i eth0
iptables -A INPUT -j ACCEPT -m state --state ESTABLISHED,RELATED
iptables -A INPUT -j ACCEPT -i ppp0 -p tcp --dport 22
iptables -A INPUT -j DROP
```

Is what I run.

Firewalling -1

```
# iptables -vL INPUT
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  lo     any     anywhere             anywhere
    0     0 ACCEPT     all  --  eth0   any     anywhere             anywhere
    0     0 ACCEPT     all  --  any    any     anywhere             anywhere            state RELATED,ESTABLISHED
    0     0 ACCEPT     tcp  --  ppp0   any     anywhere             anywhere            tcp dpt:ssh
```

Status check

What we have:

- A Linux machine running bunch of services available to LAN clients
- The same machine connected to the 'Net
- Only SSH available to the 'Net

What we don't have:

- 'Net access from other LAN machines.

DHCP - revisited

Anyone remember this?

```
option domain-name "tafe.plug.linux.org.au";
option domain-name-servers 192.168.1.1;
option subnet-mask 255.255.255.0;

default-lease-time 43200;
max-lease-time 86400;

subnet 192.168.1.0 netmask 255.255.255.0 {
  range 192.168.1.200 192.168.1.254;
}
```

Well it's missing an important line.

```
option routers 192.168.1.1;
```

This tells all the DHCP clients to use 192.168.1.1 as a router

What is a router anyway?

Simply put:
It's a machine that is connected to more than one network

and

is prepared to pass information from one network to the other.

Typical example

LAN client:

```
# ip route show
192.168.1.0/24 dev eth1  proto kernel  scope link  src 192.168.1.16
default via 192.168.1.1 dev eth1
# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth1
```

Router

```
# ip route show
202.72.191.98 dev ppp0  proto kernel  scope link  src 202.72.187.38
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.1
default via 202.72.191.98 dev ppp0
# route  -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
202.72.191.98   0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         202.72.191.98   0.0.0.0         UG    0      0        0 ppp0
```

It STILL doesn't work.!!!

Remember I said.

It's a machine that is connected to more than one network
and
is prepared to pass information from one network to the other.

Well we're obviously connected to more than one network.
What about the second part?

"Is this machine prepared to pass information
from one network to the other?"

Turning it on

look at /proc/sys/net/ipv4/ip_forward

```
# cat /proc/sys/net/ipv4/ip_forward
0
```

If the answer is "0" then this machine will NOT pass information from one network to the other

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
# cat /proc/sys/net/ipv4/ip_forward
1
```

Okay now we're all set. We're connected and we're acting as a router.
It is working right?

WRONG

About now, you're probably thinking:

Okay that's it I quit! Who need this networking thing anyway.

I'm more than happy with what I had before I came to these stinking PLUG sessions.

I'm outta here!

$#@%($@)*)Y$@!&#@!*^$#$@!@

We're almost there!

The problem is that we picked "private IP's" for our network.
It IS actually working BUT your ISP is ignoring packets from 192.168.1.16 because it doesn't have a route BACK to you.

Introducing NAT
Network Address Translation! He's our hero.

Turning IT on.

Here we go

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

What this does is mangle all of the data going out your ppp0 interface such that the "source IP" is the real IP your ISP gave you when you connected to the 'Net.

Status check

What we have:

- A Linux machine running bunch of services available to LAN clients
- The same machine connected to the 'Net
- Only SSH available to the 'Net
- It's routeing
- It's NAT'ing

Well then its working.

Limitations

Not all network protocols work.

- H323
- MSN (audio)

Some work but need helpers

- IRC (DCC sends/receives)
- FTP

Making it stick

Once again I hit the snag that each distro does things differently.

Firewall:

```
iptables        -A INPUT       -j ACCEPT     -i lo
iptables        -A INPUT       -j ACCEPT     -i eth0
iptables        -A INPUT       -j ACCEPT             -m state --state ESTABLISHED,RELATED
iptables        -A INPUT       -j ACCEPT     -i ppp0 -p tcp --dport 22
iptables        -A INPUT       -j DROP
iptables -t nat -A POSTROUTING -j MASQUERADE -o ppp0

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

Dialer:

You're okay as all the settings we changed are already saved to disk.

All you need to do is make it easy to start.

Making it stick

For both look in /etc/init.d and use the source :)

Questions

Questions?

Next meeting.

Tue, 09 Feb 2004 19:30:00 +0800