

Linux Networking Kungfu

NZ Linux Users Group Presentation
2nd February 2004

Adam Boileau
<adamb@asterisk.co.nz>

Introduction

- Welcome to Asterisk
- Adam Boileau, HOD Engineering, Asterisk
- Ex ISP packet-monkey
- Maintainer of the Asterisk Firefly embedded firewall / router / packet-mangler, a fine piece of equipment that is very reasonably priced, and doesn't have any moving parts, and forwards packets really quite well and can do all manner of blah blah blah de blah tco blah security blah Rijndael AES blah the thing with the stuff blah ...

Overview

- NAT / Mangling
- Interface Naming
- Ratelimiting / QoS
- Dynamic Routing
- Loadbalancing & Failover
- VLANs
- Policy Routing
- Problems to solve
- Tools
- Examples (x6)
- Resources
- Optional Bonus: ARP spoofing for switched sniffing, packet injection, connection hijacking, and MITM attacks on asymmetric crypto authentication.

NATting & Mangling

- Firewall tools provide ability to
 - Identify and categorise packets and packet flows
 - Modify packet headers and payload
 - Modify kernel packet metadata
 - Track connection states
- NAT table for address translation
- Mangle table for other self-manipulation
- eg. `iptables -A POSTROUTING -t mangle -j TTL -ttl-inc 1`

Interface Naming

- You can rename interfaces!
- Help identify segments, wan links, security zones
- Makes changing interfaces around easy
- Not all interfaces behave properly (eg. ppp)
- Might confuse poorly written software (eg. Iptraf)
- eg. `ip link set eth0 name internal; ip link set eth1 name dmz; ip link set eth2 name external`

Ratelimiting & QoS

- Control flow rate of packets leaving the host
- Control the order in which packets leave the host
- Can be complicated – packet classifiers, queuing disciplines, filters, command line tools not friendly. Parts by Chinese PhDs, other bits by Russian nuclear scientists.
- Userland tool `tc` (from `iproute2`)
- Kernel HTB
- Good howto at <http://www.lartc.org>

Dynamic Routing

- Allows router to modify it's routing tables
- Learn routes from neighbors
- Routing protocols provide authentication, loop control, and handle changing topography, congestion
- Available on Linux:
 - BGPv4 (bird, zebra)
 - OSPF (bird, zebra)
 - RIP (routed, bird, zebra)

Load Balancing & Failover

- Aggregate multiple interfaces together (bonding)
 - For bandwidth (when 10GBE isn't enough)
 - For layer 2 redundancy
 - 802.3ad (Cisco LACP)
- Detect failures, alter routes, interfaces (heartbeat)
- Loadbalancing requests to servers (LVS, NAT)
- Firewall failover
 - State replication and “loose” state pickup

802.1q VLANs and trunking

- Split up a switch's ports into separate segments
- Isolate different types of traffic (eg. VOIP, incessantly noisy broadcast prone protocols)
- Connect VLANs together (ISL Trunking)
- Linux talks trunking – one interface per VLAN
- Build dynamic, multi-interface firewalls that'll do thousands of interfaces at hundreds of megabits
- Kernel VLAN support, userland tool `vconfig`

Policy Routing I: Normal Routing

- Routing based on destination only
- One global route table
- Nasty old `route` tool
- Shiny new `ip` tool (`iproute2`)
- Acceptable for boring edge hosts ...
- ... if your mummy tucks you up in bed in your flannel Basil Brush PJs with a glass of warm milk.

Policy Routing II

- Routing based on things other than destination
 - Source address
 - Packet header info (eg. TOS bits, 14 ports)
 - Packet metadata (eg. fwmark)
 - External factors (eg. Link cost or utilization)
- More than one routing table
- Kernel Advanced Routing
- Userland tool `ip`

Summary of Tools

- 2.4 Kernel network components
- Oldschool `route`, `ifconfig` and `arp`
- `iproute2`'s `ip` and `tc`
- `iptables`
- `tcpdump`
- `vconfig`
- Packet generators (eg. `hping2`, `nemesis`, `scapy`)

Problems you can solve

- Implement cost-policies (Telecom DSL)
- Implement asymmetric routing (eg. Starnet)
- Transit provider network (eg. Walker Wireless)
- Perform transparent proxying (eg. Squid)
- Handle conflicting addressing (eg. Impossible VPNs)
- Implement neat-o features (eg. FireflyX2 Phonehome)

DSL Cost Mitigation

- DSL Circuit, and an analogue modem
- 56k modem = 14GB/month = \$14k (on full rate DSL)
- Route important packets via DSL, bulk, all traffic while at work and in bed, and all traffic after cap exceeded via modem:
 - Firewall rule to count packets, cron job to check count and add rule to mark all packets for the modem if cap exceeded
 - Firewall rules to mark full-service packets that always get DSL (eg. Quake)
 - `iptables -A PREROUTING -t mangle -m tos --tos minimise-cost -j MARK --set-mark $MARK_MODEM`
 - `iptables -A PREROUTING -t mangle -m time -timestart 08:30 -timestop 17:30 -days Mon,Tue,Wed,Thu,Fri -j MARK --set-mark $MARK_MODEM`
 - `ip rule add table modem fwmark $MARK_MODEM`
 - `ip route add table modem default dev ppp0`
 - `ip route add default via $DSL_GW_IP`

Asymmetric Routing: Pest's Network

- Oldschool Starnet (4mbps) ingress, 128k DSL egress
- Egress traffic spoofs a source address routed back via starnet
- VPN to colo-host to tunnel out spoofed traffic because of Telecom IPNet's reverse-path filtering
- `iptables -A POSTROUTING -t nat -o tap0 -j SNAT --to-source $STARNET_IP`
- `ip route add $COLO_PUB_IP dev ppp0`
- `ip route add default via $COLO_VPN_IP dev tap0`

Transit Provider Network

- Walker Wireless provide transit network for ISPs to sell bandwidth
- Address space allocated by ISPs
- No global default route; packets must go from customer to their ISP to get to the internet
 - `for net in $WW_ICONZ_PREFIXEN; do`
 - `ip rule add from $net table iconz;`
`done`
 - `ip route add table iconz default via $ICONZ_GW`

Transparent Proxying

- Transparently pass all 80/tcp traffic through a proxy
- Locally: iptables -A PREROUTING -t nat -p tcp --dport 80 -j REDIRECT--redirect-to 8080
- To another machine:
 - iptables -A PREROUTING -t nat -p tcp --dport 80 -j DNAT --to-destination \$PROXY:8080
 - If the proxy and the source are on the same network: iptables -A POSTROUTING -t nat -d \$PROXY -j SNAT --to-source \$OUR_IP
- To another machine, preserving source address:
 - ip tables -A PREROUTING -t mangle -p tcp --dport 80 -s !\$PROXY -j MARK --set-mark 1
 - ip rule add table proxy fwmark 1
 - ip route add table proxy default via \$PROXY
 - On proxy: REDIRECT as above

Conflicting Addressing: No one can defeat the Quad-NAT

- Two customers want a VPN between them. Both are 192.168.10.0/24.
- Customer B wants to talk only MSRDP to Customer A's box 192.168.10.10, and only from source 192.168.10.156.
- B has a route to 10.0.0.1/32 via ipsec to A
- `iptables -A PREROUTING -t nat -s 192.168.10.156 -d 10.0.0.1 -i ipsec0 -j DNAT --to-destination 192.168.10.10`
- `iptables -A POSTROUTING -t nat -s 192.168.10.156 -d 192.168.10.10 -j SNAT --to-source 10.0.0.2`
- `iptables -A PREROUTING -t mangle -s 192.168.10.10 -d 10.0.0.2 -j MARK --set-mark 0xbeef`
- `iptables -A PREROUTING -t nat -s 192.168.10.10 -d 10.0.0.2 -j DNAT --to-destination 192.168.10.156`
- `ip rule add fwmark 48879 table argh`
- `ip route add table argh 192.168.10.156 via $IPSEC_GW dev ipsec0`
- `iptables -A POSTROUTING -t nat -s 192.168.10.10 -d 192.168.10.156 -j SNAT --to-source 10.0.0.1`

FireflyX2 Phonehome

- FireflyX2 thin client wants to phone home (when the user specifically asks!) from unknown network
- Makes ssh connection on port 443/tcp to Bastien at Asterisk, with DSA key.
- Sets up PPP tunnel over SSH connection, authenticates with it's serial number, and is allocated an address from a dynamic pool
- Bastien adds a NAT rule to convert dynamic address to the FX2's allocated static address, for which we have internal DNS.
- Bastien emails the engineering department to advise us who's connected.
- Engineer connects to serialnumber.customer, provides support.
- Bastien generates ICMP-net-unreach for the FX2-static network, so packets for unconnected hosts dont get in a routing loop:

```
ip route add unreachable $FX2STATICNET
```

Resources

- www.netfilter.org
- www.lartc.org
- www.faqs.org/docs/iptables
- snafu.freedom.org/linux2.2
- “Linux Routing” by Dee Ann LeBlanc, Joe "Zonker" Brockmeier, Ronald W. McCarty Jr.
ISBN: 1578702674
- www.policyrouting.org

ARP Spoofing

- Injecting fake ARP “is at” messages, to cause switched traffic to come to our host
- Sniffing in a switched environment
- Injecting packets (eg. TCP RST)
- Man-in-the-Middle attacks on crypto protocols
- Live connection hijacking with TCP-sequence fix-up
- Protection: switch port security, static arp entries, both of which are almost impossible to maintain.
- If the hax0r gets on your LAN, fj34r. :)