Xurble

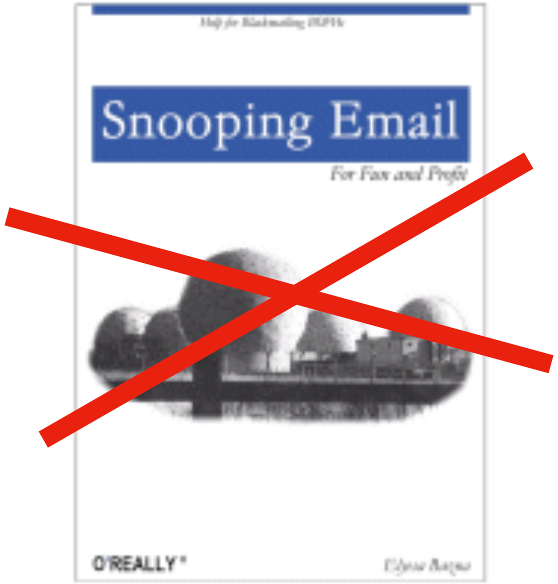
conntrack, Netfilter, NetFlow and NAT under Linux

Oliver Gorwits

9th February 2010
Milton Keynes Perl Mongers

“Policy Compliance”

- We have legal obligations
- Avoiding the courts ✔
- Avoiding the newspapers ✔



(alleged) Copyright Violations

Subject: File-sharing of unauthorised content owned by Twentieth Century Fox
From: fox_eve_p2p-no-reply@copyright-compliance.com

Dear Oxford University:

Twentieth Century Fox Film Corporation, located in Los Angeles, and its affiliated companies (collectively, 'Fox') own intellectual property rights, including exclusive rights protected under copyright laws, in many motion pictures, television programs and other audio-visual works, including the motion picture AVATAR (collectively, the 'Fox Titles').

Fox conducted an online check by scanning public networks and discovered that your Oxford University internet account was used to access and distribute an unauthorised copy of AVATAR. By distributing Fox content without Fox's permission, you infringed Fox's copyright. Here is the information Fox obtained from the online check:

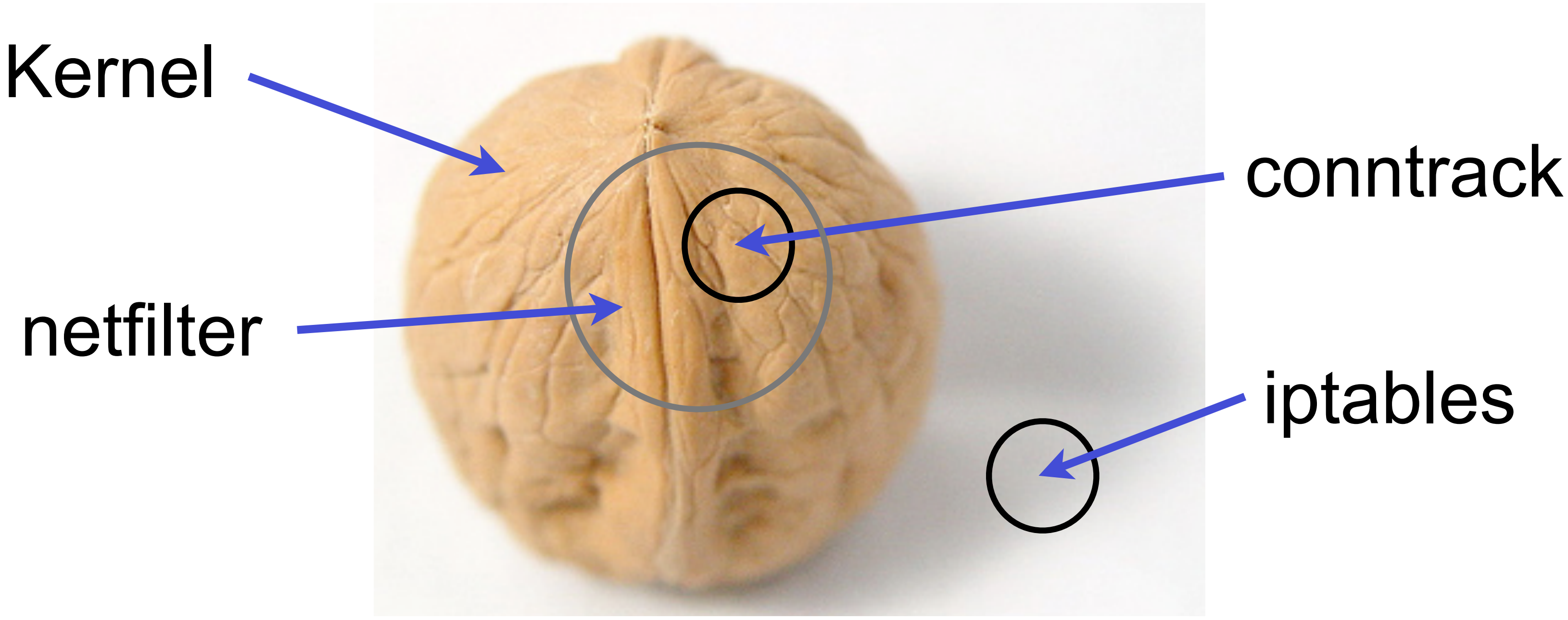
Timestamp of report: 07 Feb 2010 23:12:44 GMT
Title details: Avatar (2009) PROPER TS XviD-MAXSPEED
IP address: 163.1.xxx.yyy
Port ID: 30854
Protocol used: BitTorrent - L5

Please respond to Fox and **identify what steps you have taken to resolve this matter** by contacting Fox at fox_eve_p2p@copyright-compliance.com

The Process

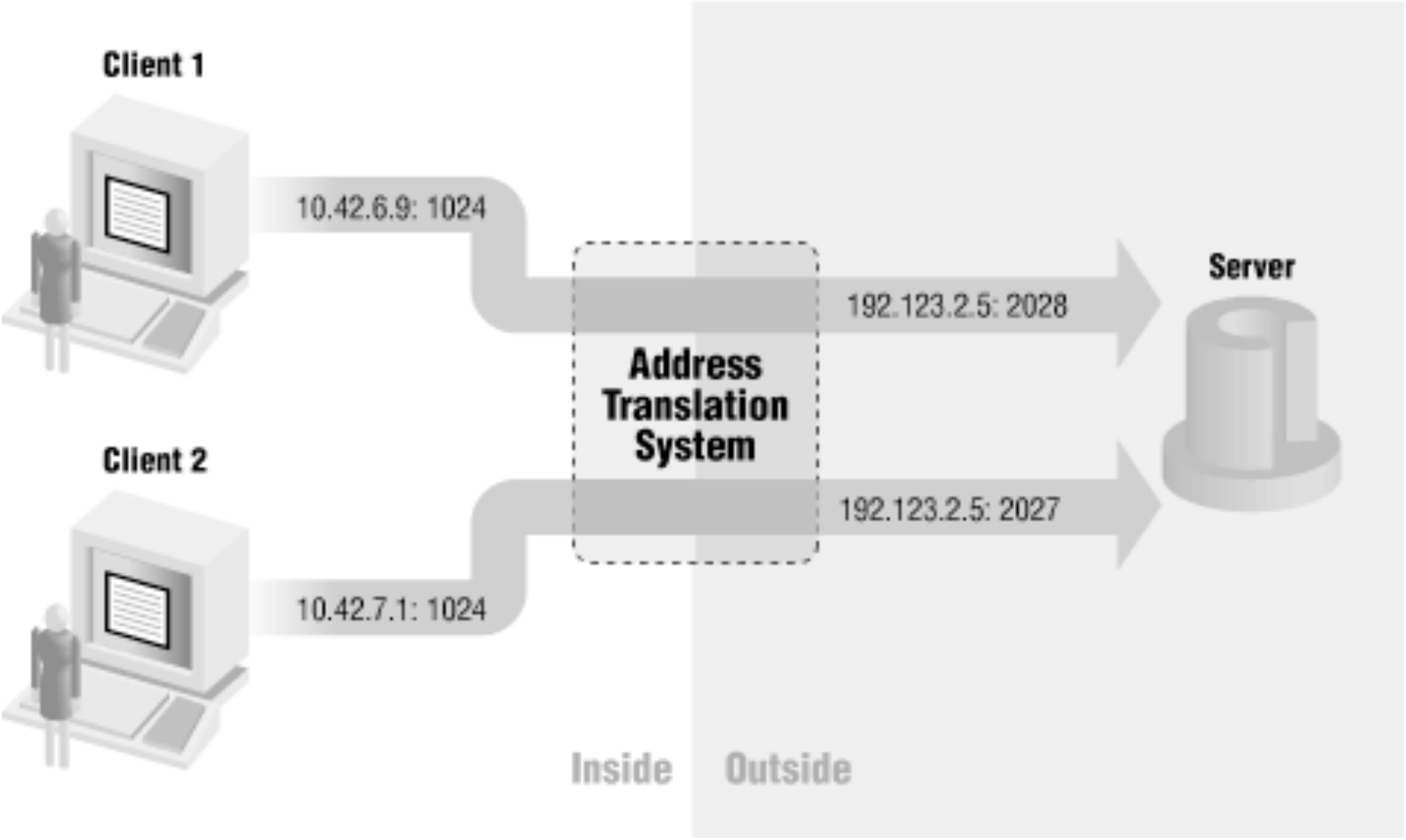
- So, given:
  - ○Timestamp with Time Zone
  - ○IP address
  - ○***TCP port number***
- We need:
  - ○User’s identity
- Usually via:
  - ○Network log-in logs, and DHCP logs

Linux network subsystems



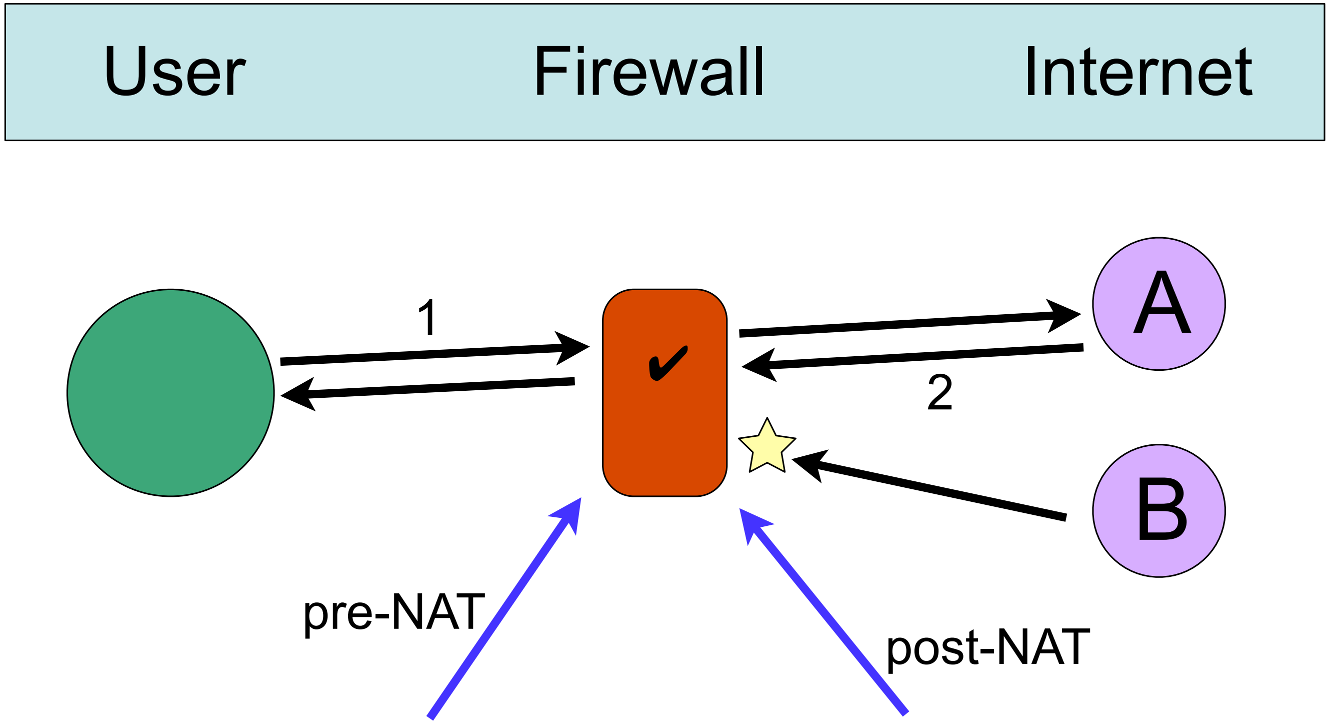
pietroizzo

Network Address/Port Translation



O'Reilly

State Tracking



User Firewall Internet

1

✔

2

A

B

pre-NAT

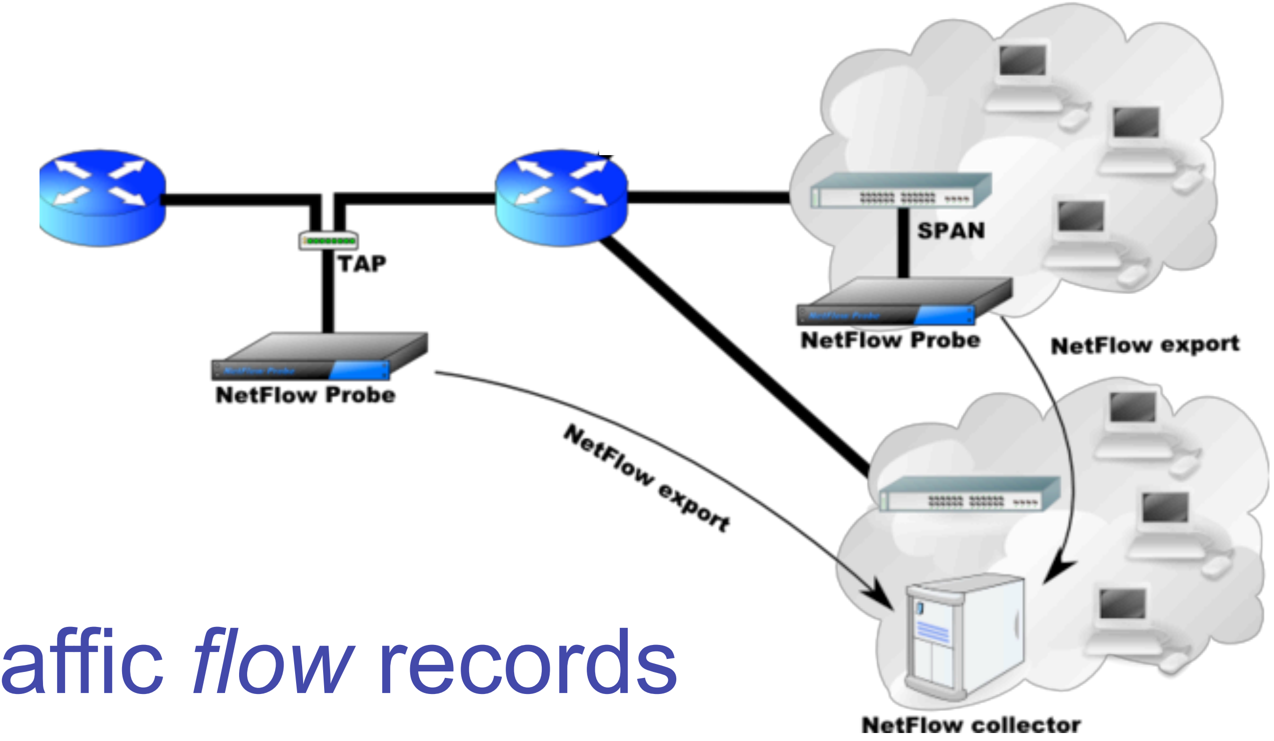
post-NAT

- Traditional loggers run two packet captures and correlate the timestamps.
- This is processor intensive, and cannot be 100% accurate.

Conntrack CLI tool

- “Userspace tool to search, list, inspect and maintain the connection tracking subsystem of the Linux kernel”
- “Monitor connection tracking events, e.g. **show an event message (one line) per newly established connection**”
- Uses kernel’s ctnetlink interface, so efficient

NetFlow



- Export traffic *flow* records
- Seven-key tuple (src, dst, IP, ports, etc)
- Many tools freely available
- Used for capacity planning, security

From conntrack to NetFlow

- Following the Unix pattern of CLI filter programs, translate conntrack into NetFlow

```
[NEW]     tcp 6 120 SYN_SENT src=10.16.207.250 dst=66.249.91.104
sport=50199 dport=80 [UNREPLIED] src=66.249.91.104 dst=192.76.7.253
sport=80 dport=43243

[DESTROY] tcp 6 src=10.16.207.250 dst=74.125.79.104 sport=50140 dport=80
packets=7 bytes=2006 src=74.125.79.104 dst=192.76.7.253 sport=80
dport=61284 packets=7 bytes=4083
```

becomes...

```
1229348772,28728000,585028,163.1.3.221,1,84,584929,585028,10.16.207.250,18.7
.22.83,0.0.0.0,0,0,0,0,1,0,0

1229348772,28728000,585028,163.1.3.221,1,84,584929,585028,192.76.7.254,18.7.
22.83,0.0.0.0,0,0,0,0,1,0,0
```

Perl: IPC::Run

- system() and background procs w/ piping, redirs, ptys (Unix, Win32)
- “a module that can handle full Bourne shell pipe syntax internally, with fork and exec, without ever invoking a shell.”

```
# MAIN BODY
    run $conntrack,
        '|', \&conntrack2flowtools,
        '|', sub { ptee($config) },
        '|', $flow_import,
        '|', $flow_send;
```

Perl: Daemon::Generic

- A framework for starting, stopping, reconfiguring daemon-like programs

```
use Daemon::Generic;

sub gd_run { ... stuff }
sub gd_preconfig { ... stuff }

# hand control to Daemon::Generic
# control is returned through callbacks
newdaemon();
```

```
Usage: $progname [ -c file ] [ -f ]
                 { start | stop | reload | restart | help | version | check }
```

Linux: daemontools

- A collection of free tools for managing Unix services as an improvement to the inittab, ttys, init.d or rc.local alternatives
- Nowadays probably replaced by `upstart`

```
\_ supervise nfflowd
   \_ /usr/bin/perl /usr/bin/nfflowd -f start
       \_ /usr/sbin/conntrack -E -e NEW,DESTROY -o timestamp -n
       \_ /usr/bin/perl /usr/bin/nfflowd -f start
       \_ /usr/bin/perl /usr/bin/nfflowd -f start
       \_ /usr/bin/flow-import -z0 -f2 -V5 -m0xFF31EF
       \_ /usr/bin/flow-send -V5 -s 192.0.2.1/203.0.113.1/21001
```

Net::Netfilter::NetFlow

- http://search.cpan.org/perldoc?Net::Netfilter::NetFlow
- “Generate a stream of Cisco NetFlow logging data for all TCP, UDP and ICMP connections passing through.”
- No reconfiguration of the firewall ruleset or network is required
- Efficient and accurate, using Netfilter's own connection logging and tracking data

Amarand Agasi

Q&A

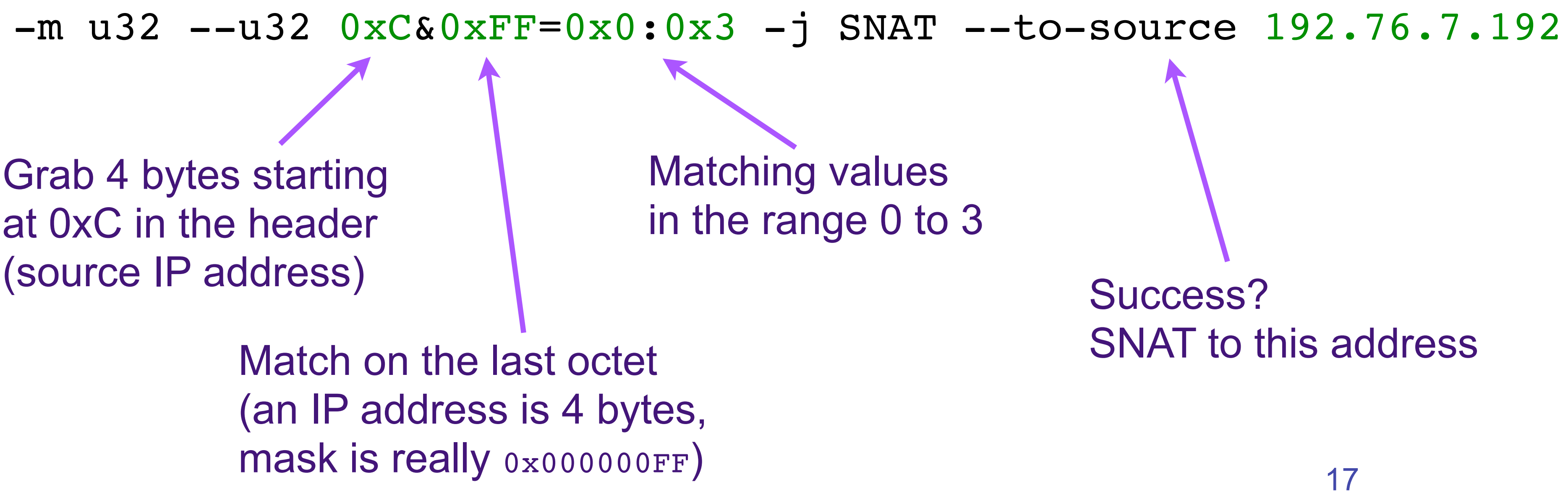
Bonus slides:
u32-based NAT balancing

- 192.76.7.192/26 ➔ 64 outside IPs
- 10.0.0.0/14 ➔ ~256k possible client IPs
- Only some parts of 10/14 are in use
- Want to ensure balanced use of outside IPs
- Netfilter u32 module to the rescue!
- Inspect some bytes from the IP header

u32 NAT balancing implementation

- Map last octet of client IP to an outside IP

```
a.b.c.0  ⎫
a.b.c.1  ⎬    192.76.7.192
a.b.c.2  ⎪
a.b.c.3  ⎭
```



```
-m u32 --u32 0xC&0xFF=0x0:0x3 -j SNAT --to-source 192.76.7.192
```

Grab 4 bytes starting at 0xC in the header (source IP address)

Match on the last octet (an IP address is 4 bytes, mask is really `0x000000FF`)

Matching values in the range 0 to 3

Success? SNAT to this address

schmilblick

http://www.flickr.com/groups/computer-plates/