OpenBSD in the wild ...
...a personal journey

Avik Sengupta
Chief Technology Officer
Itellix Software Solutions Pvt Ltd

Agenda

- OpenBSD – Why and How ...
- ... In the context of our experiences!

The context

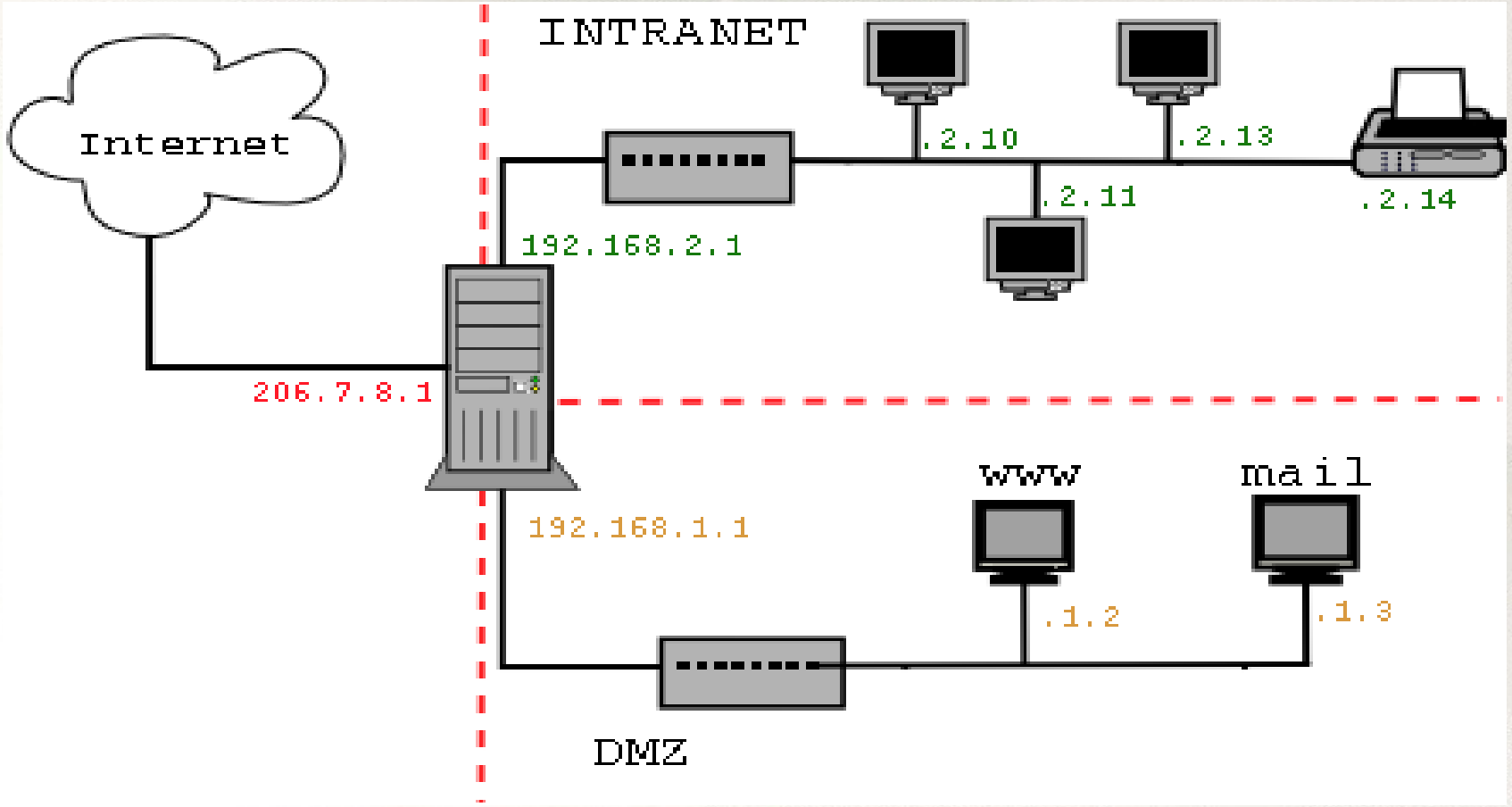
- Startup, circa 2002 AD
- No sysadmin, limited budgets

  BUT
- No Compromises in the network
  - On Security
  - On Functionality

OpenBSD Basics

- Derived from 4.4 BSD
- Proactive security
  - Extensive source code audits
  - Integrated cryptography
  - Minimalist default install
- “Only one remote hole in the default install in more than 10 years!”
  - That was 7 the last time there was a session on OpenBSD in this conference :)
- Highly portable (i386,sparc,ppc,hppa ...etc..)
- Free, under a BSD Licence

Our Network Topology



Services on our OpenBSD gateway

- Internet Router
  - Bandwidth Monitor
- Firewall
  - NAT
  - Port Redirector
- VPN gateway
- Mail Gateway
  - Spam Check
  - Virus Check
- Caching HTTP proxy

Getting Started

- Just follow the installation instructions
  - My second ever OBSD install went to production!
    - And stayed there!
  - http://www.openbsd.org/faq/faq4.html
  - Only relatively complex issue is disk partitioning
  - Good hardware detection
    - But dont try bleeding edge hardware

OS, not a distribution

- Out of the box
  - Dev Environment (gcc 3.3.5 with propolice)
  - SSH server
  - Ntp, dhcp, named (DNS), cvs, apache (chrooted)
  - Advanced Networking: BGP, OSPF, RARP
  - Gateway services: ftp_proxy, tftp_proxy, spamd
  - Security services: Kerberos, IPSec

Minimal Default Install

```
# ps ax
  PID TT   STAT      TIME COMMAND
    1 ??   Is      0:00.09 /sbin/init
30262 ??   Is      0:00.01 syslogd: [priv] (syslogd)
10426 ??   I       0:01.98 syslogd -a /var/empty/dev/log
26637 ??   Is      0:00.01 inetd
18647 ??   Is      0:06.69 /usr/sbin/sshd
 6744 ??   Is      0:00.53 cron
21297 ??   Is      0:04.99 sendmail: accepting connections (sendmail)
17432 ??   Is      0:00.01 (unlinkd) (unlinkd)
10892 ??   Is      0:00.28 sshd: root@ttyp0 (sshd)
19693 ??   Is      0:00.01 comsat
22714 C2   Is+     0:00.01 /usr/libexec/getty Pc ttyC2
32253 C3   Is+     0:00.01 /usr/libexec/getty Pc ttyC3
15937 C5   Is+     0:00.01 /usr/libexec/getty Pc ttyC5
#
```

Ports and Packages

- Packages are pre-compiled binaries, architecture specific (think RPM)
- Ports are customised makefiles, used to build software from source (think Gentoo)
- Packages are quicker, Ports are more flexible

Customising the install

- Customise daemons in rc.conf
  - Enable NTP
- Install essential packages
  - Squid - cached http proxy
  - Postfix – easier to configure compared to sendmail
    - SpamAssasin
    - ClamAV
    - Anomy Sanitizer
  - Snmpd – monitor services using snmp
  - OpenVPN – SSL VPN gateway, simplest to setup

Packet Filter (pf)

- Introduced in 2001 in OpenBSD 3.0
- Filter TCP/IP traffic and perform Network Address Translation
- Intercept each IP packet, passing or blocking it
- Stateless inspection, based on fields in each packet
- Statefull inspection, keeping track of connections
- Packet Normalisation

pf Rules

- Our biggest reason to use OpenBSD
  - Looked at other ruleset options
  - Most intuitive rule syntax
  - Easy to understand even for a non-expert
- Rules contain parameters that match a packet, and then block or pass it
- Rules can create state
  - Represents an established connection
  - Keyed on 4-tuple: source{ip,port} and dest{ip,port}

Example IPTables script

```
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP
$IPTABLES -N bad_tcp_packets
$IPTABLES -N allowed
$IPTABLES -N tcp_packets
$IPTABLES -N udp_packets
$IPTABLES -N icmp_packets
$IPTABLES -A bad_tcp_packets -p tcp --tcp-flags SYN,ACK SYN,ACK \
-m state --state NEW -j REJECT --reject-with tcp-reset
$IPTABLES -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j LOG \
--log-prefix "New not syn:"
$IPTABLES -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j DROP
$IPTABLES -A allowed -p TCP --syn -j ACCEPT
$IPTABLES -A allowed -p TCP -m state --state ESTABLISHED,RELATED -j
    ACCEPT
$IPTABLES -A allowed -p TCP -j DROP
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 21 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 22 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 80 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 113 -j allowed
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j ACCEPT
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 11 -j ACCEPT
IPTABLES -A INPUT -p tcp -j bad_tcp_packets
$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -s $LAN_IP_RANGE -j
    ACCEPT
```

```
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LO_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LAN_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $INET_IP -j ACCEPT
$IPTABLES -A INPUT -p UDP -i $LAN_IFACE --dport 67 --sport 68 -j
    ACCEPT
$IPTABLES -A INPUT -p ALL -d $INET_IP -m state --state
    ESTABLISHED,RELATED \
-j ACCEPT
$IPTABLES -A INPUT -p TCP -i $INET_IFACE -j tcp_packets
$IPTABLES -A INPUT -p UDP -i $INET_IFACE -j udp_packets
$IPTABLES -A INPUT -p ICMP -i $INET_IFACE -j icmp_packets
$IPTABLES -A INPUT -m limit --limit 3/minute --limit-burst 3 -j LOG \
--log-level DEBUG --log-prefix "IPT INPUT packet died: "
$IPTABLES -A FORWARD -p tcp -j bad_tcp_packets
$IPTABLES -A FORWARD -i $LAN_IFACE -j ACCEPT
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j
    ACCEPT
$IPTABLES -A FORWARD -m limit --limit 3/minute --limit-burst 3 -j LOG \
--log-level DEBUG --log-prefix "IPT FORWARD packet died: "
$IPTABLES -A OUTPUT -p tcp -j bad_tcp_packets
$IPTABLES -A OUTPUT -p ALL -s $LO_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $LAN_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $INET_IP -j ACCEPT
$IPTABLES -A OUTPUT -m limit --limit 3/minute --limit-burst 3 -j LOG \
--log-level DEBUG --log-prefix "IPT OUTPUT packet died: "
$IPTABLES -A POSTROUTING -o $INET_IFACE -j SNAT --to-source
    $INET_IP
```

Equivalent PF script

```
set skip on $LO_IFACE
nat on egress inet from $LAN_IFACE:network
  to any -> (egress)
scrub in
block log
pass out inet
pass in on (egress) inet proto tcp from any
  to (self) port { 21, 22, 80, 113 }
pass in on (egress) inet proto icmp from
  any to (self) icmp-type echo-req
pass in on $LAN_IFACE inet from
  $LAN_IFACE:network modulate state
```

Advanced PF features

- Queues and prioritisation
  - Bandwidth reservations
- Routing (eg. Using multiple uplinks)
- Anchors and Tables for dynamic ruleset changes
- Integration with application level proxies
  - Ftp, tftp, transparent squid
- All features work with Ipv6
- Logging (pcap/tcpdump compatible)

Upgrades

- Watch for critical patches
  - http://www.openbsd.org/errata.html
- Patch Strategies
  - Pick and choose patches or Follow -stable
  - Kernel compile from src in either case
- Binary upgrade supported only on consecutive release (3.9 -> 4)
  - Can't skip a release
  - Ocassional format and install is our preferred option
- Read the FAQs!!

Questions?

Curiosity killed the cat... or did it?

Thank You!

- Resources
  - http://www.openbsd.org/
  - http://www.openbsd.org/faq/index.html
  - man pf; man pf.conf; man pfctl
  - http://www.openbsd.org/cgi-bin/man.cgi
  - http://www.undeadly.org/
  - This presentation:
    - http://www.sengupta.net/talks/

Final (two) words

- FAQ
- man