OSCAR SECURITY

Neil Gorsuch
NCSA

The Futility of Security

- Even extreme measures don't always work
- Cases in point:
  - Airport security
  - Recent openssh security hole
- CERT security incidents
  - (each may involve a single machine to hundreds of machines)

Year	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
Incidents	252	406	773	1,334	2,340	2,412	2,573	2,134	3,734	9,859*

Year	2000	Q1-Q3, 2001
Incidents	21,756	34,754

OSCAR Security Difficulties

- Wide range of security hardening requirements
  - Some clusters have all nodes directly connected to internet
  - Some already behind massive firewalls
- Every cluster machine needs hardening in some cases
- Widely varying cluster network topologies
- Clusters usually have many network services
- Clusters usually have many users
- Clusters often need to be accessible from the internet
- Clusters are prime targets for hackers:
  - Powerful computing engines
  - Large temporary data storage
  - Fast network for launching denial of service attacks

The Loose Security Environment/Model

In political terms, the FAR LEFT:

- All cluster nodes accessible from the internet
- No firewalls between the cluster and the internet
- Little or no packet filtering
- No high ports filtered from the internet
- All network services allowed
  - (what, you mean RedHat isn't secure out of the box?)

The technical name for this type of environment is a

The Loose Security Environment/Model

In political terms, the FAR LEFT:

- All cluster nodes accessible from the internet
- No firewalls between the cluster and the internet
- Little or no packet filtering
- No high ports filtered from the internet
- All network services allowed
  - (what, you mean RedHat isn't secure out of the box?)

The technical name for this type of environment is a "target rich environment"

The Loose Security Environment/Model

In political terms, the FAR LEFT:

- All cluster nodes accessible from the internet
- No firewalls between the cluster and the internet
- Little or no packet filtering
- No high ports filtered from the internet
- All network services allowed
  - (what, you mean RedHat isn’t secure out of the box?)

The technical name for this type of environment is a “target rich environment” or “script kiddies playground”

The Strict Security Model (FAR RIGHT)

- Very strict cluster/border firewall
  - No packet forwarding, all services must use proxies
  - Sometimes, secure login services such as ssh not allowed
    - (because they can’t log conversations)
- Access outside the firewall to the cluster impaired
- File transfer difficulties

The technical term for this is

The Strict Security Model (FAR RIGHT)

- **Very strict cluster/border firewall**
  - No packet forwarding, all services must use proxies
  - Sometimes, secure login services such as ssh not allowed
    - (because they can’t log conversations)
- **Access outside the firewall to the cluster impaired**
- **File transfer difficulties**

  The technical term for this is “work impairment”

- **“Big Brother” mentality**

  (logs of all your actions)

"I SEE YOU'VE DISCOVERED OUR 'FIREWALL'"

OSCAR Security Plans

- Machine based packet filtering for exposed nodes
  - (yes/no during wizard, and a configuration GUI)
- TCPwrapping
  - (yes/no during wizard, and a configuration GUI)
- Network services control GUI
- Sandboxes around servers if wanted
  - (ftp, http, DNS, NTP, etc.)
- Examination of logs for hacking attempts
- Tripwire if wanted
- IDS if wanted

Security is an Onion (Layers)

- Router packet filtering (outside of OSCAR’s scope)
- Source route verification (OSCAR 1.?)
- Machine packet filtering (OSCAR 1.?)
  - At externally connected nodes if cluster has private subnet
  - On all cluster nodes if no private subnet
  - Hard to set up unless special tools available
  - Easy to mess up and leave security holes without special tools
- TCPwrappers (OSCAR 1.?)
  - Conditionally block access by service/source address
  - Some services bypass this in some distributions
- Server application sandboxes (OSCAR 1.?)
- Application configuration (OSCAR 1.?)

  Some applications can be set to conditionally block access
- Users cannot log into compute/batch nodes (OSCAR 1.?)
- Intrusion Detection Services (OSCAR 1.?)
  - Tripwire, others, some sniff network packets

Source Route Verification

- When turned on for an interface, prevents packets from being accepted on that interface that are impossible (such as packets claiming to come from 127.0.0.1 arriving on a NIC)
- Martians are packets that are dropped by source route verification
- Planned to be in OSCAR 1.3/1.4 as part of pfilter

Packet filtering

- Examines each network packet, looking at:
  - Source address abd destination addresses
  - Network interface
  - Protocol type (TCP UDP ICMP) and destination port
  - Other parameters depending on type
- Each packet is either:
  - Passed
  - Dropped (nasty, makes port scanning more difficult, and makes harmless wrong connection attempts take a long time to timeout)
  - Rejected (normal network protocol for packet that can’t be accepted)
- Two types of packet filters are stateless and stateful
- Planned to be in OSCAR 1.3/1.4 as part of pfilter

Blocking Outside Access To Services

- **Best way to tighten security**

  (most network break-ins through service security holes)

- **Difficult way to tighten security**

  (different distributions turn on/off services different ways)

  (different services conditionally block access different ways)

- **TCPwrappers can block/filter most services**

  /etc/hosts.allow and /etc/hosts.deny

Best Security Return on Invested Time

- **Packet filtering**
  - Does the work of most of the other security layers
  - Hard to set up
  - Easy to mess up
  - Some network services require special connection tracking modules not all of which are written yet

- **What is needed:**

  A good way to generate packet filtering rulesets that is:
  - Easy to use
  - Shields the user from packet filtering details
  - (a packet filtering compiler)

pfilter – a Packet Filter Compiler

- Configured by either:
  - An easy to understand text configuration file, or …
  - a configuration GUI (both hide details of packet filtering)
- Default setup:
  - Incoming packets/connections that are not explicitly allowed are dropped/rejected
  - Any outside network service can be accessed/used from within the machine with no proxies
  - Incoming ICMP packets blocked except for:
    - Destination unreachable and redirect packets for errors
    - OSCAR defaults will allow pinging and tracrouting within cluster
- Controlled by chkconfig/service start stop restart status
- No compiled binaries, same RPM for all platforms

pfilter on a Multiple NIC Machine

- Automatically determines which is the protected internal interface by looking for RFC 1918 private addresses
- By default:
  - Allows machines on the internal/protected interface to access any outside network service
  - Hides machines on the internal interface from the outside
  - Uses IP masquerading for all connections from the inside
- Allows internal hosts to have some services be accessible from the outside using translated address/ports. These hosts can appear to respond to a separate external address than the one that pfilter is running on. This allows hosts on a private subnet behind the firewall/pfilter machine to not have to be tightened down, security-wise, while still allowing access to them from the outside.

Main Parts of a pfilter Configuration File

This opens/closes services on the pfilter machine:

```
OPEN/CLOSE [protocol(s)] service(s) port(s)  \
   [from source(s)] [to destination(s)] [on interface(s)]
```

This allows hidden hosts on a private subnet to appear to be outside the firewall machine with selected services on the hosts being accessible from outside:

```
ALIAS  external_address hidden_address [protocol(s)] \
   service(s) port(s) [from source(s)] [on interface(s)]
```

This is how to set network interface attributes:

```
filtered|unfiltered|protected|unprotected interface(s)
```

pfilter Ruleset Generation

- Ruleset generation is done when started/restarted
- Generates “glue” rules that (if appropriate):
  - Turn on IP masquerading and packet forwarding
  - Generate pseudo interfaces for aliased hosts
  - Add routing table entries for aliased hosts
  - Logs bad packets various ways
  - Uses modprobe commands to pull in kernel modules
  - Turn on source route verification and martian logging
  - Turn off BOOTP packet forwarding
  - Allow packets that are part of a tracked connection through
- Allows/blocks specified connections

pfilter Efficiency

A 3 line configuration file required to:

- Allow ssh logins to a machine
- Allow ssh logins to one of it’s internal hidden hosts
- Allow terminal services logins to another hidden host

produces a good level security 192 line ruleset file

pfilter, A Work in Progress

- Version 1 is in use:
  - on largest NCSA cluster
  - in various NCSA X-in-a-box products
- Open sourced (GNU license)
- Resides on sourceforge
- Version 2 is in testing stages

pfilter Version 2

- Currently being tested
- New features:
  - Network services defined in text files, extensible by user
  - Services can include shell script fragments for OPEN/CLOSE
  - Macros in configuration and service definitions
  - Support for iptables/ipchains/ipfilter through macros
  - Conditional “compilation” in configuration and services
  - Named constants/defines in configuration and services

pfilter Future Versions

- GUI (webmin based)
- TCPwrappers <> pfilter synchronization
- System logs analysis for hacking attempts reporting

For pfilter Information

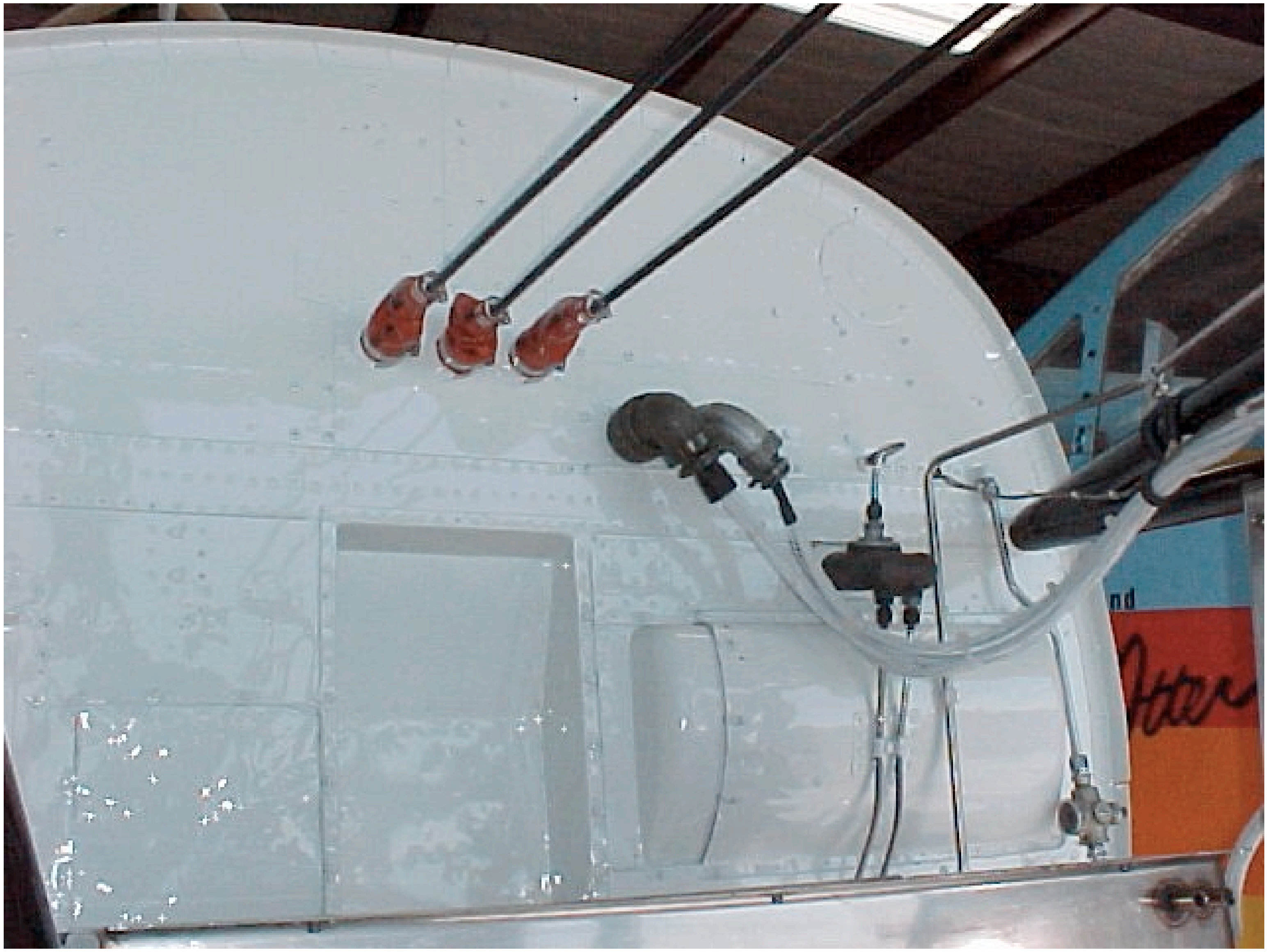
- Contact me at:
  ngorsuch@ncsa.uiuc.edu
- pfilter project page
  - http://pfilter.sourceforge.net/

The Perfect Firewall for OSCAR 2.0

- **A firewall was found that:**
  - Does more than filter packets
  - Completely blocks all unwanted intrusions through it
  - Lets exactly what the user wants through it
  - Rock – solid
  - Proven technology

- **I’m still trying to integrate it into OSCAR, but here is an advance look at it before it’s deployment:**

The PERFECT Firewall



Private Subnet Cluster Topology

- Only head(login)/administration nodes on outside network
- Head(login)/administration nodes have dual NICs
  - One to outside network, other to private cluster network
- Compute/storage nodes on private network
- Compute/storage nodes access outside network through the head/login/administration nodes
- Compute/storage nodes are NAT’ed/IP masqueraded
- More difficult to allow access to compute nodes from outside
  - (too bad about CACTUS)
- Better security for entire cluster
- Relaxed security concerns on compute/storage nodes
- Throughput problems from compute/storage to outside
  - (probably not usable for medium or larger clusters)

Public Network Cluster Topology

- Simpler to set up
- Allows direct access to compute nodes from outside
  - (can be UN-filtered to specific nodes from specific network addresses via job control language when needed - CACTUS)
- Worse overall cluster security
- ALL cluster nodes need packet filtering and/or security tightening
- ALL cluster nodes are potential targets
- Better network throughput
- Large clusters probably need to be set up this way

Stateless Packet Filtering

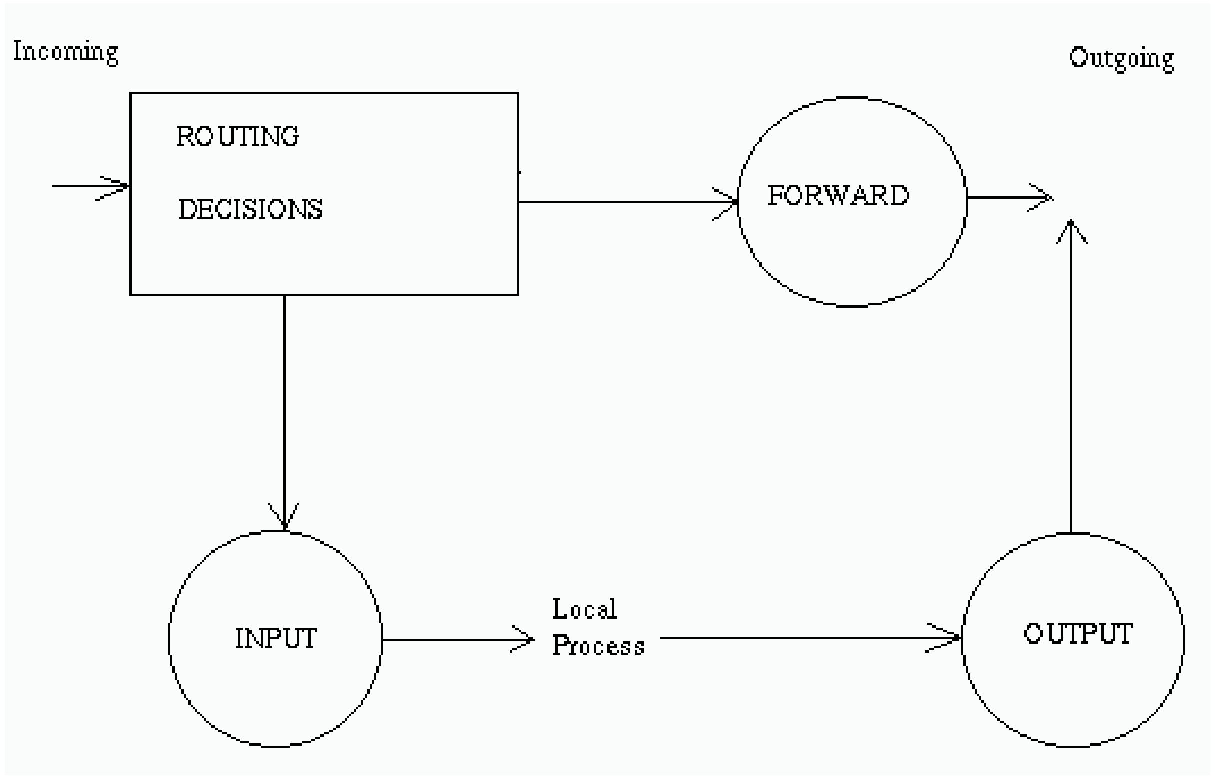
- Linux kernels prior to 2.4
- Controlled by the ipchains program
- Examines each packet on it’s own to pass/drop/reject
- Does not keep track of running connections
- Because of this, high ports usually have to be left unblocked
- FTP is a big problem under stateless packet filtering
- Typical stateless ruleset 5 times larger than for stateful
- Much less security than stateful packet filtering
- Fortune 500 example

Stateful Packet Filtering

- Linux kernels 2.4 and later
- Controlled by the iptables program
- Keeps track of active connections, examines each packet based on their place in a connection
- Can easily block ALL ports except the ones needed to connect to services
- Can conditionally unblock other ports based on their appropriateness in an active connection
- Rulesets can be very simple and still allow good security

Linux kernel >=2.4 / iptables Filtering

- Pre-defined filtering:



- Integrated connection tracking / stateful filtering / NAT
- New conditional packet matching: MAC Address, limits to prevent log overflows, user id, packet length, packet strings, time of day, service classifications

Login Services

- **telnet**
  - Most send names/passwords over network un-encrypted
  - Newer kerberized versions avoid this and are acceptable
- **rlogin**
  - Names/passwords over network uncrypted (unless user .rhosts files allowed)
  - Anything rlogin can do, ssh can do better
- **ssh**
  - Encrypted connections
  - Kerberized versions available
  - Openssh has excellent track record
  - Allows tunneled X connections (X ports can be blocked)

File Transfer Services

- tftp
  - Simpler version of FTP
  - Security Nightmare
- ftp
  - Widely used
  - Most send names/passwords over network un-encrypted
  - Server security holes found frequently
  - Nightmare for stateless packet filtering
  - Better to force users to passive mode ftp from within cluster
- scp
  - Encrypted connections
  - Kerberized versions available

FTP Active Mode

Sequence:

- Client program contacts server on TCP port 21
- Client picks TCP port 1024-65535 on client for the server to connect to
- Server connects back to the client chosen port coming from TCP port 20

Active mode FTP clients on the inside of a firewall should only be allowed to access the outside if:

the firewall has stateful packet filtering so that it can selectively open higher TCP ports based on current FTP connections.

Active mode FTP servers inside a firewall are not a problem for either stateless or stateful packet filter.

FTP Passive Mode

Sequence:

– Client program contacts server on TCP port 21
– Server picks an unused TCP port 1024-65535 and tells the client the port number
– Client opens connection to picked TCP port on server

Clients outside a firewall should only be allowed to access FTP servers inside the firewall in passive mode if:

the firewall has stateful packet filtering so that it can selectively open higher TCP ports based on current FTP connections.

Clients inside a firewall can access outside servers in passive mode without any problems for stateless or stateful packet filtering.

Remote Windowing

- X
  - The standard for Linux/Unix
  - Serious security holes
  - Don’t allow it through a firewall unless tunneled inside ssh
  - Linux/Unix/Windoze clients available
- Windoze Terminal Services
  - Too bad it’s not usable for Linux, because it’s encrypted, compressed, detachable, and caches bitmaps
- VNC http://www.uk.research.att.com/vnc/index.html
  - Stateless (reconnect sessions from anywhere)
  - Sharable (sharing of window sessions)
  - Encrypted passwords, un-encrypted stream (tunnel via ssh)
  - Servers/clients for Unix/Linux/Windoze/MAC/X