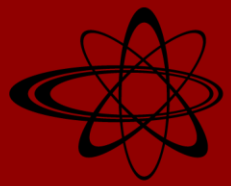


SNRG
Security and Networks Research Group

FIREWALL RULE SET OPTIMIZATION

Author Name: Mungole Mukupa
Supervisor : Mr Barry Irwin [CISSP]
Security and Networks Research Group
Department of Computer Science
Rhodes University

Overview

- Project background
- Chosen firewall
- Rule set development
- Performance Benchmarking
- Optimizing [How]
- Result graphing
- Optimizer design
- Conclusion

Project Background

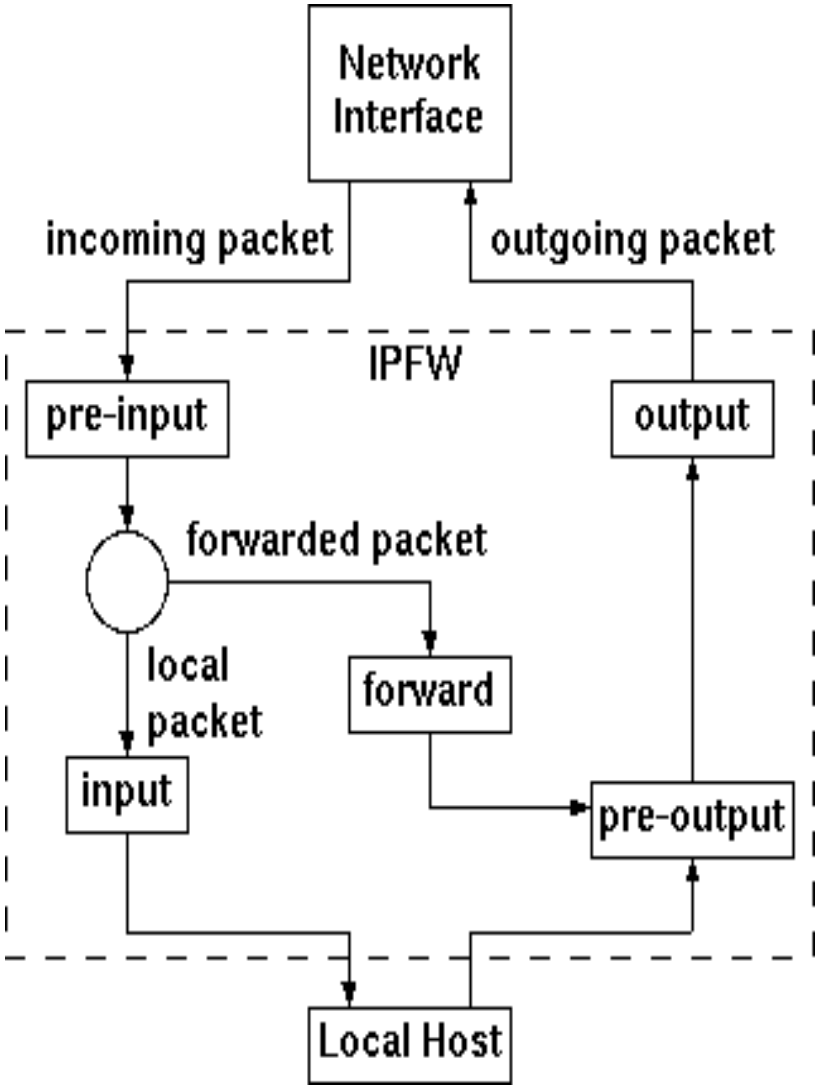
- A firewall is configured to filter traffic based on set rules developed from the security policy.
- It is evident that NOT all rules are matched in a rule set
- This degrades firewall performance as all rules in set are checked against a packet that wont match it.
- The foregoing plus the nature of network traffic being dynamic is the motivation for this project.
- The project is focusing on filtering at the Network Layer of the TCP/IP stack

```
   I   /firewall.test                Row 40   Col 1   10:25  Ctrl-X H for help
$cmd add 0001  allow ip from any to any via lo0
$cmd add 0002  deny ip from any to 127.0.0.0/8
$cmd add 0003  deny ip from 127.0.0.0/8 to any
$cmd add 0004  check-state

#####################################
# Close out RFC1918 nets: No spoofing
#####################################
$cmd add 0005 deny all from 10.0.0.0/8 to ${ip}
$cmd add 0006 deny all from 172.16.0.0/12 to ${ip}
$cmd add 0007 deny all from 192.168.0.0/16 to ${ip}

#####################################
# Allow HTTP traffic
#####################################
$cmd add 0008 allow tcp from any to ${ip} 80 setup in keep-state
$cmd add 0009 allow ip from any to ${ip} 443
#####################################
# DNS
#####################################
$cmd add 0010 allow udp from any to ${ip} 53 in keep-state
$cmd add 0010 allow tcp from any to ${ip} 53 setup in keep-state
$cmd add 0010 allow udp from ${ip} to any 53 keep-state
#####################################
```

Chosen Firewall

- Unix FreeBSD IPFW is being used as the development and testing firewall
- Open source
- Support for IPv6 (ipfw8), stateful and stateless processing of connections.
- BSD kernel offers better integration with compiler programs like C++
- widely used in operating systems. Apple's Mac pcs and Microsoft now has wipfw [windows IP Firewall]
- Features for traffic shaping, Bandwidth control and NAT for IP aliasing – increase throughput

FreeBSD IP FIREWALL [IPFW]



Current Focus : Rule Set Development

- Rules have been written and continue being revised

- IPFW types as set in FreeBSD's rc.conf are being tested to note the filtering differences.

- Packet capture tools have been installed – Tcpdum, Wireshark[dumppcap], Ettercap. Tools perform differently hance the variation.

- Testing tools being used [TCPReplay,Bittwist, Etherreal]

```
^[ (escape) menu  ^y search prompt  ^k delete line   ^p prev li   ^g prev page
^o ascii code     ^x search         ^l undelete line ^n next li   ^v next page
^u end of file    ^a begin of line  ^w delete word   ^b back 1 char
^t begin of file  ^e end of line    ^r restore word  ^f forward 1 char
^c command        ^d delete char    ^j undelete char ^z next word
L: 49 C: 38 ================================================================
# server
cmd="ipfw -q add"
ipfw -q -f flush
ks="keep-state"


#loopback
$cmd 10 allow all from any to any via lo0
$cmd 20 deny all from any to 127.0.0.0/8
$cmd 30 deny all from 127.0.0.0/8 to any
$cmd 40 deny tcp from any to any frag

# stateful
$cmd 50 check-state
$cmd 60 allow tcp from any to any established
$cmd 70 allow all from any to any out keep-state
$cmd 80 allow icmp from any to any

# services

#ftp
$cmd 110 allow tcp from any to any 21 in
$cmd 120 allow tcp from any to any 21 out
#ssh
$cmd 130 allow tcp from any to any 22 in
$cmd 140 allow tcp from any to any 22 out
#smtp
$cmd 150 allow tcp from any to any 25 in
$cmd 160 allow tcp from any to any 25 out
#dns
$cmd 170 allow udp from any to any 53 in
$cmd 175 allow tcp from any to any 53 in
$cmd 180 allow udp from any to any 53 out
$cmd 185 allow tcp from any to any 53 out
#http
$cmd 200 allow tcp from any to any 80 in
$cmd 210 allow tcp from any to any 80 out
#pop3
$cmd 220 allow tcp from any to any 110 in
$cmd 230 allow tcp from any to any 110 out
#ntp
$cmd 240 allow udp from any to any 123 in
$cmd 250 allow udp from any to any 123 out
#https
$cmd 260 allow tcp from any to any 443 in
$cmd 270 allow tcp from any to any 443 out

# deny log
$cmd 999 deny log all from any to any
```

Performance Benchmarking

- IP throughput testing is being used. to measure:
  - Maximum Transfer rate
  - Rule depth matching and count rate
  - Connection Rate [a device inspecting using more rules takes longer to accept new connections]
  - Connection Tear Down
  - Illegal Traffic Handling [is denied traffic being allowed through?]
  - Legal Traffic Handling [are there rules overshadowing allowed packets?]
  - Connection establishment
  - Latency Test: determine and measure latency for stateful and stateless protocols.

Optimizing

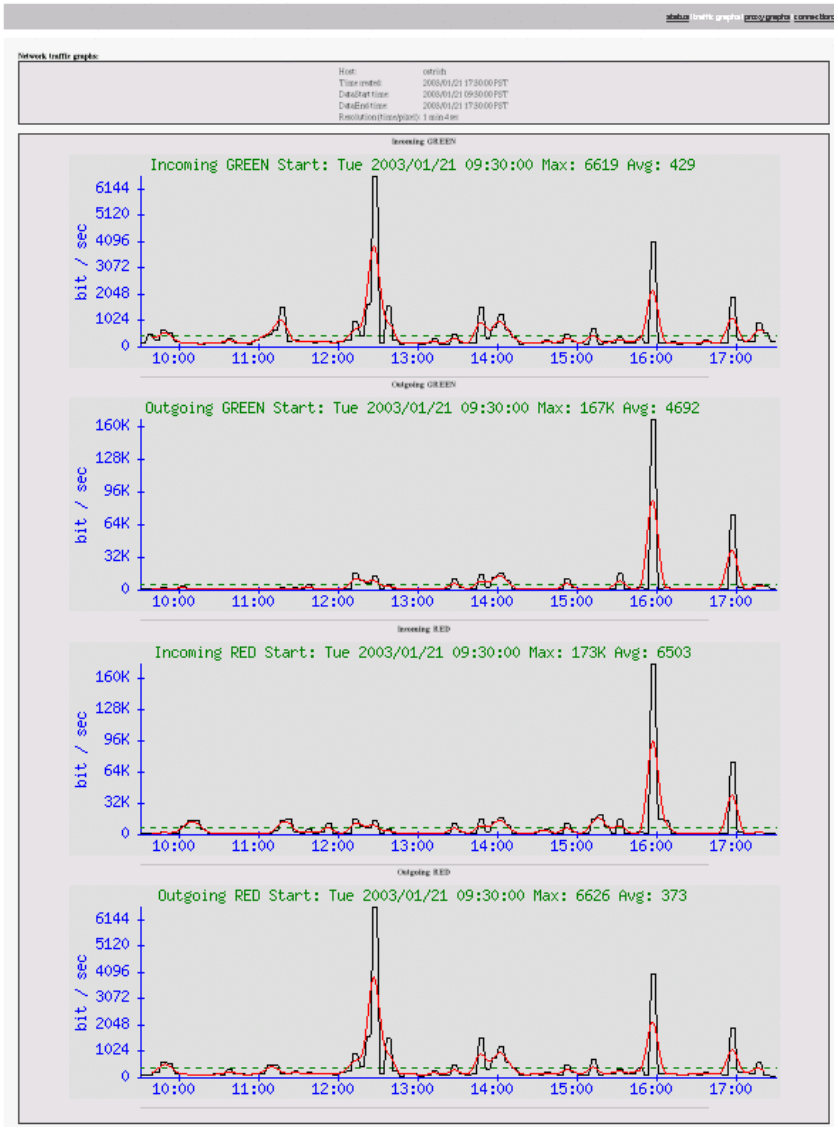
- The tests are being done using a rule set not as long as those in production networks for experimental reasons.

- The optimization is based on
  - match counts per rule.
  - Hit to trigger ratio [rule is inspected but may not match]
  - Rule ranking [priority]
  - Other values being observed during testing.

```
^[ (escape) menu  ^y search prompt  ^k delete line   ^p prev li   ^g prev page
^o ascii code     ^x search         ^l undelete line ^n next li   ^v next page
^u end of file    ^a begin of line  ^w delete word   ^b back 1 char
^t begin of file  ^e end of line    ^r restore word  ^f forward 1 char
^c command        ^d delete char    ^j undelete char ^z next word
L: 49 C: 38 ============================================================
# server
cmd="ipfw -q add"
ipfw -q -f flush
ks="keep-state"


#loopback
$cmd 10 allow all from any to any via lo0
$cmd 20 deny all from any to 127.0.0.0/8
$cmd 30 deny all from 127.0.0.0/8 to any
$cmd 40 deny tcp from any to any frag

# stateful
$cmd 50 check-state
$cmd 60 allow tcp from any to any established
$cmd 70 allow all from any to any out keep-state
$cmd 80 allow icmp from any to any

# services

#ftp
$cmd 110 allow tcp from any to any 21 in
$cmd 120 allow tcp from any to any 21 out
#ssh
$cmd 130 allow tcp from any to any 22 in
$cmd 140 allow tcp from any to any 22 out
#smtp
$cmd 150 allow tcp from any to any 25 in
$cmd 160 allow tcp from any to any 25 out
#dns
$cmd 170 allow udp from any to any 53 in
$cmd 175 allow tcp from any to any 53 in
$cmd 180 allow udp from any to any 53 out
$cmd 185 allow tcp from any to any 53 out
#http
$cmd 200 allow tcp from any to any 80 in
$cmd 210 allow tcp from any to any 80 out
#pop3
$cmd 220 allow tcp from any to any 110 in
$cmd 230 allow tcp from any to any 110 out
#ntp
$cmd 240 allow udp from any to any 123 in
$cmd 250 allow udp from any to any 123 out
#https
$cmd 260 allow tcp from any to any 443 in
$cmd 270 allow tcp from any to any 443 out

# deny log
$cmd 999 deny log all from any to any
```

Results Graphing

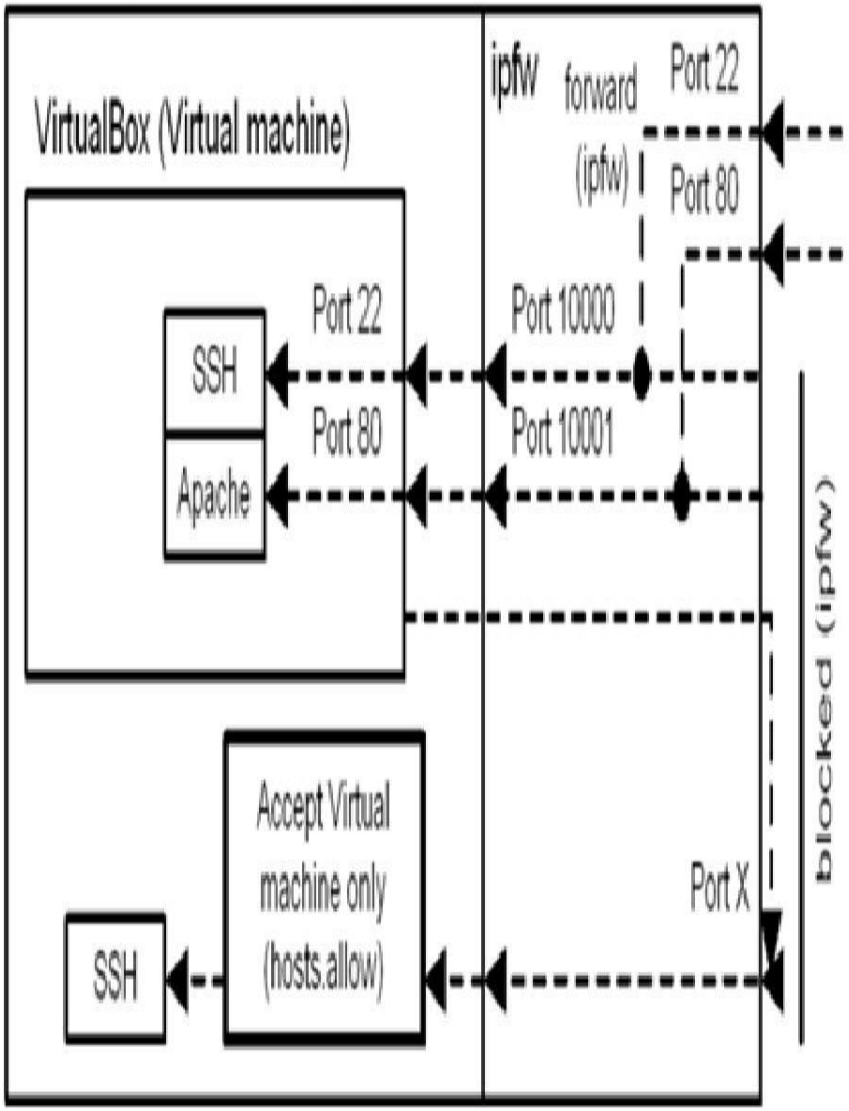
- Firewall traffic treatment will be reported in graphical form to show the improvement in inspection time-**throughput**
- The default [un-optimized] rule set is being tested and results graphed
- Tests will be repeated on the optimized rule set using the same traffic and results graphed
- These results will then be compared and reported for drawing the conclusion
- Tools are being used to aid in real time traffic plotting - Gnuplot, ipfw-graph, MRTG



Current Focus: Optimizer Design

- This will take an integration of various optimizing techniques

- Some techniques under consideration are :
  - Rule recency : when last was a rule matched and what traffic caused the firing of that rule?
  - Hit ratio : How often a rule is matched after inspection
  - Rule merging:/Aggregation: write one rule for rules seen to be matching the same traffic
  - Rule reordering: moving frequently matched rules to the beginning of the rule set
- The design will be focused more on once performance tests have been done.
- Most previous research reviewed has found this to be **NP-HARD** but we cant continue folding our arms.



Conclusion

More in due course !

Thankyou for attending

Additions, Clarifications , Subtractions ?