



How to Improve your Firewall Performance

Prof. Avishai Wool
AlgoSec CTO & Co-Founder
and
Tel Aviv University

- The Problem
- Static Analysis
- Usage-based Analysis
- Rule Reordering
- Demo



Does Your Firewall Look like this?



Causes for Clutter in Firewall Rules

- Dynamic business environment leads to constant rate of rule changes (dozens of changes per week!)
- Multiple administrators, Staff turnover
- Outsourced firewall management (Managed Security Service Providers)
- Results: “Configuration Clutter”
 - Check Point Firewall with 1,200 rules and 15,000 objects
 - PIX configuration with 50,000 lines



Results of Configuration Clutter

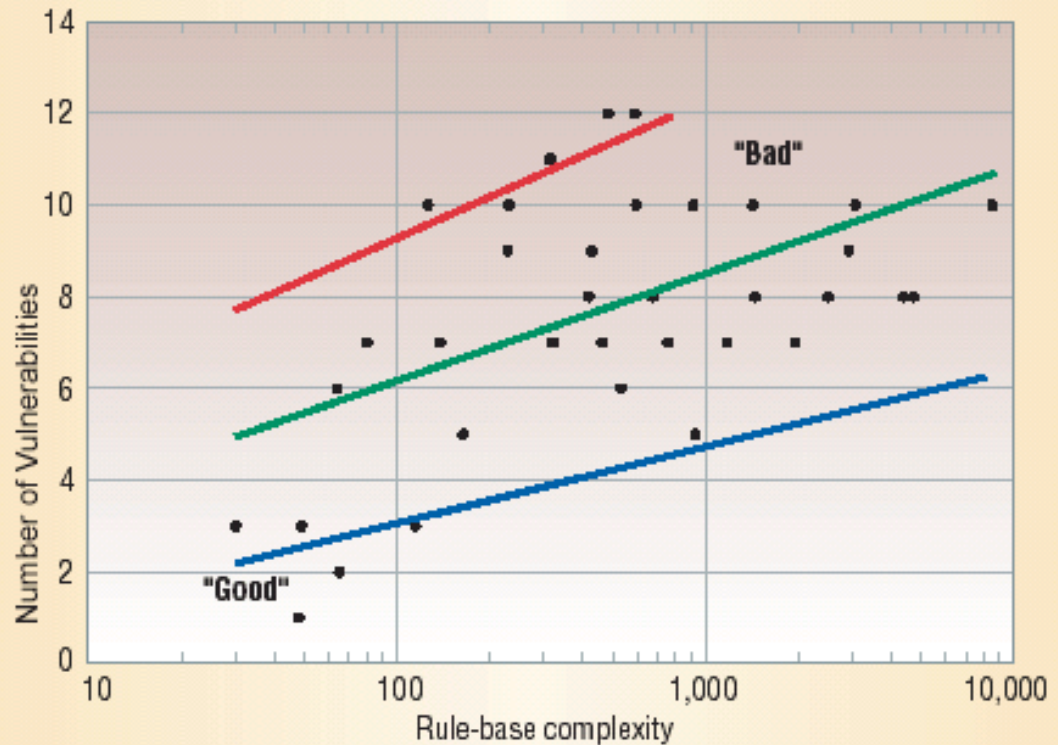
- Firewall slows down – may become a bottleneck
- Hardware size limitations – may need bigger hardware
- Slow and cumbersome management interface
- But also...

Configuration Clutter is a Security Risk



A survey of the firewall policies of 30 US-based large corporations suggest that more complex policies are more exposed to serious risk.

Source: IEEE Computer magazine, June 2004



$$\text{Rule-base complexity} = \text{Rules} + \text{Objects} + (\text{Interfaces})^{**2}$$

Cleanup Phase 1: Static Analysis

Some definitions are clearly useless

- Unattached Object / Unattached VPN user group:
An object that
 - does not appear in any rule
 - every group it belongs to does not appear in any rule
 - In any policy on any firewall

- Empty Objects:
 - Do not refer to any IP address

- Unattached VPN Users:
 - Do not appear in any user group → have no access

- Unattached access-list (Cisco)
 - Not connected to any interface



- Expired VPN users
 - No longer have access

- Disabled Rules:
 - Maybe it's time to delete them?

- Time-Inactive rules:
 - Timed Rules are active on a certain days of the month, days of the week, or times of the day...
 - ... But you cannot set a year.
 - Identify the expired rules before they will become active again next year.

- Vendor tools cannot answer the question
 - “does this definition already exist with another name?”

- Result:
 - Administrators often define the same object (Host, Subnet, or Group) multiple times

- Difficult to find without software assistance



- Firewalls process the rules in-order (“first match”)
- If “early” rules match every packet that a “late” rule could match – the “late” rule is covered (== useless!)
- Easy cases:
 - single rule covers another rule
 - the object names match exactly
- Most vendors don’t check. One vendor can catch simple cases.



Covered Rules – In-depth analysis

- The combination of rules 15 & 33 covers rule 35
- Need to dive into the definitions !
- Requires some algorithmics to discover

RULE	SOURCE	DESTINATION	SERVICE	ACTION
15	GP_NW_SLI_LAN	GP_NW_Garden_ICN	TCP http TCP https TCP ldap	accept
33	GP_NW_BAI_LAN	GP_NW_Garden_ICN	* Any	accept
35	GP_NW_BAI_LAN GP_NW_SLI_LAN	NW_Garden_ICN_003	TCP https	accept

Name is different – but all IP addresses are covered !

Cleanup Phase 2: Usage-based Analysis



- Over time:
 - Applications are discontinued
 - Servers are relocated to other IP addresses
 - Test environments move to production
 - Business partnerships change
 - Networks are re-architected
 - Routing is changed

- Result: Firewalls still have the rules –
 - but the traffic is gone

- Idea: flag rules and objects that have not been used “recently”



- Firewalls can log each matched packet
- Log includes rule number, timestamp, and more
- Naïve approach: Filter the logs based on rule number and find the missing numbers

- Challenge #1: Logging is configured per rule
 - Some rules don't produce logs

- Solution #1: List rules that do not produce logs separately

Challenge #2: Unreliable rule numbers



NO.	NAME	SOURCE	DESTINATION	VPN	SERVICE	ACTION	TRACK
traffic rules (Rules 7-18)							
7	Req #1882	algo_cl1 algo_cl2 FakeNet	algo_cl1 lab_internal cpmodule	All_GwToGw	TCP http TCP ssh TCP A12345 TCP CP_Exnet_resolve	accept	- Non
8	Req #1895	WebServers	yabbba_1	Any Traffic	TCP sqlnet1	accept	- Non
9		algo_cma	IP sub_inside	Any Traffic	TCP daytime-tcp	drop	- Non
10		Any	ALGO-CL algo_cma dis_test_host	All_GwToGw	TCP telnet UDP archie	accept	Log

Which rule was “rule 9” on March 11?

Unreliable rule numbers - solutions



- Solution #2: Vendor attaches a unique “rule_id” to each rule, such that:
 - Reported to log
 - Remains with rule through rule add/remove/modify

- Check Point – feature available starting with NGX R60
 - Consider an upgrade...

- Cisco – feature available starting with PIX/ASA v7.x

- Juniper Netscreen: unique rule_id exists, but not reported to log!
 - Need to work around this limitation...

Usage Information: hit counters



- Cisco Firewalls & Routers maintain a per-rule hit-counter

- Advantages:
 - Unrelated to logging: un-logged rules are counted too
 - Rule insertions & deletions do not affect the hit-counters

- Challenge:
 - hit-counters are reset to zero when device reboots

- Solution:
 - Take periodic snapshots
 - Attach pseudo rule_uids, homogenize the snapshots
 - Make sure not to double-count ...

Maintain a long history



- Some rules only work occasionally or rarely
 - High-shopping season
 - Disaster recovery rules – tested semi-annually
- Need usage information of many months

- Challenge:
 - Log files can become huge
 - Logs are discarded or rotated
 - Hit-counters are occasionally set to 0

- Solution:
 - Process the raw usage information –
 - ... But only keep concise summaries (forever)



Now that we have usage information

- Unused Rules:
 - have not matched traffic in the last NNN days

- Unused Objects:
 - Do not belong to any rule that matched traffic in the last NNN days

- Most / Least used rules

- Last date that rule was used
 - Even if it is listed as “unused”

Rule Reordering



- Reminder: Firewalls process the rules in-order
- Once a connection is matched – firewall moves to next connection

- Amount of CPU time spent depends on position of matching rule in the rule-base:
 - 1st rule: firewall only tests one rule – minimal CPU
 - Last rule: firewall tests all the rules – high CPU

- Reordering approach (first attempt):
Move the popular rules to the top

Not every reordering is acceptable

- Again: Firewalls process the rules in-order
- If a rule is moved to the top – it may supersede (and contradict) the policy!

- Ridiculous Example:
 - Last rule in the rule-base is the default
“from any to any with any service, drop”
 - Moving it to position 1 will drop all traffic (very efficiently)

- Reordering approach (better):

Move the popular rules as high as possible without changing the firewall policy

A new firewall performance model



- RMPP – Rules Matched Per Packet
- Intuition: calculate the average number of rules the firewall tested until it reached the rule that matched a connection
- Average is taken with respect to usage statistics



- Example 1: Policy of only one rule:
 - (Deny all or permit all)
 - RMPP = 1

- Example 2: Policy of 100 rules:
 - Usage: rule #1: 20%, rule #10: 30%, rule #100: 50%:
 - $RMPP = 1 \cdot 20\% + 10 \cdot 30\% + 100 \cdot 50\% = 53.2$
 - On average, the firewall tests 53.2 rules per connection



Arrange the rules to minimize the RMPP, without changing the firewall policy

- Note:
 - RMPP is a model
 - It ignores some causes for CPU utilization
 - Your mileage may vary...



- Reordering 100's of rules via point-and-click is unrealistic
- But: Often a very small number of rule moves produces a very big change in RMPP
- Solution:
 - Suggest Top-10 most effective rule movements



Meet AlgoSec at Booth D198

Email: avishai.wool@algosec.com

Thank You!

Supported Platforms



- Check Point FireWall-1 (all versions)
 - Sun Solaris ✓ Linux ✓ Win-NT ✓ Nokia ✓
 - SecurePlatform ✓ Alteon ✓ NSF ✓ Provider-1 ✓
 - Crossbeam ✓ OPSEC integration ✓
- Cisco PIX (all versions) ✓
- Cisco FWSM ✓
- Cisco ASA ✓
- Cisco IOS Router Access Control Lists ✓
- Juniper Firewalls (NetScreen) ✓





- If we switch the relative order of any “PASS” rule and any “DROP” rule – we may change the policy
 - Not always

- A Safe approach:
 - Split the rule-base into alternating blocks of PASS and DROP rules
 - Only allow movement of rules inside blocks – never from block to block

The full reordering algorithm



- Split the rule-base into alternating blocks of PASS and DROP rules
- Order the rules to minimize the RMPP separately within each block



- The RMPP is an imperfect model.
- It ignores:
 - Un-logged rules
 - Later packets of a long session only cause a quick “state lookup”, not a full rule-base search
 - CPU cycles spent on “deep packet inspection”
 - CPU cycles spent on VPN encryption/decryption
 - Effects of hardware accelerators
- So: An 60% improvement in RMPP may yield a smaller improvement in CPU utilization