Parallel Firewall Designs for High-Speed Networks

Ryan J. Farley

WAKE FOREST UNIVERSITY
Computer Science
Network Security Group
nsg.cs.wfu.edu

US Department of Energy
Office of Science
U.S. DEPARTMENT OF ENERGY
MISC Division

Computer Science MS Defense • Fall 2005

Abstract

- Firewalls are vital to security policy enforcement
- However, they introduce significant delay to a system
- What will happen in the next generation of networks?
- This presentation will introduce a novel parallel firewall system
- Objects:
  - Maintain Quality of Service
  - Mitigate Denial of Service
  - Provide High Scalability

Modern Security Issues

- Connections to the Internet can leave a network vulnerable
- Conventionally a firewall is utilized like a router, between a group of networks
- Not just a routing table, they enforce an ordered set of rules
- Called a **security policy**, or ACL
- Knowledge of previous decisions is **state**

Example Policy Representations

- **Best match** vs Last match vs First match
- Tree/Graph methods show that input style may vary from actual implementation

1 Deny all traffic
2 Allow traffic from host x with any service
3 Deny traffic from any host with service y

Figure 1: Example Psuedo-policy with "all traffic" rule at top

Example Policy Representations

- Best match vs **Last match** vs First match
- Tree/Graph methods show that input style may vary from actual implementation

1 Deny all traffic
2 Allow traffic from host x with any service
3 Deny traffic from any host with service y

Figure 2: Example Psuedo-policy with "all traffic" rule at top

Example Policy Representations

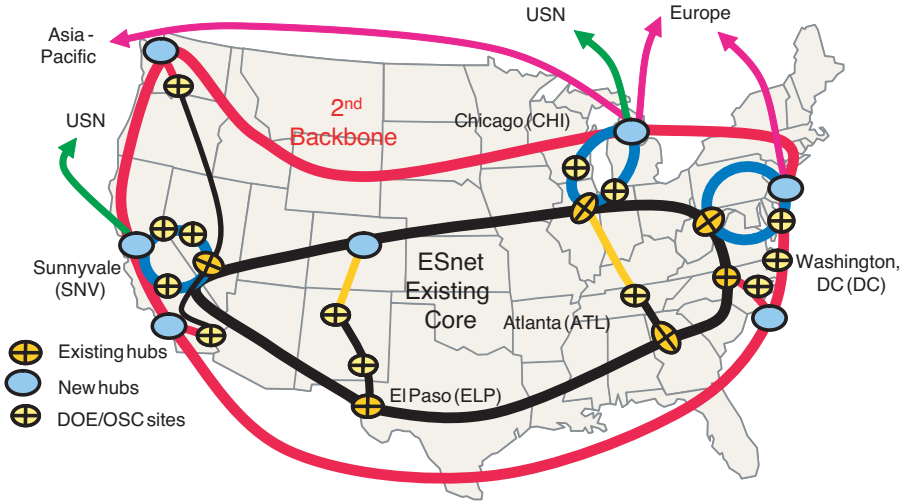
- Best-match vs Last-match vs **First-match**
- Tree/Graph methods show that input style may vary from actual implementation

1 Allow traffic from host x with any service
2 Deny traffic from any host with service y
3 Deny all traffic

Figure 3: Example Psuedo-policy with "all traffic" rule at bottom

ESnet and UltraNet

- DOE network to support climate analysis and simulation
  - Facilities are located across the United States
- Network consists of leased fiber (OC 192) and Gigabit Ethernet
  - Maximum data rate is 5 Gbps



- Several **important** security issues are present

Allowing for High Speed Networks

- Security policy enforcement imposes significantly higher processing loads than routing
- This will only increase as networking technology advances
- Several solutions for improving firewall performance
  1. Optimize algorithms
  2. Optimize rules
  3. Parallelize system
- Rule optimization is an area of future research (Matt Lane)
- Improvements for a single firewall can be made, but are a temporary solution

A Candidate for Parallelization

- Firewalls are a candidate for parallelism
- Two types:
  1. **Data parallel (DP)** – divides data processed
  2. **Function parallel (FP)** – divides work of processing data
- Data parallel
  - Scalable to load
  - Fails to reduce policy processing time
- Function parallel
  - Reduces policy processing time
  - Allows higher performance capabilities

What I Will Cover Today

- Background Material (Policy Concepts)
- Current Approaches
- Function Parallel Design
  - With Gate
  - With no Gate
- Theoretical Layout
- Simulation Results
- How to DIY

Firewall Modeling Concepts

- A rule is an ordered tuple and an associated action

$$r = (r[1], r[2], \ldots, r[k])$$

- Any tuple of a rule can be fully specified or contain wildcards '*'
- A packet is the same but has neither ranges nor an action

$$d = (d[1], d[2], \ldots, d[k])$$

- **Definition** Packet d matches r_i if

$$d \Rightarrow r_i \quad \text{iff} \quad d[l] \subseteq r_i[l], \quad l = 1, \ldots, k$$

Policy Models

- A firewall enforces a **policy**

  Definition A **policy** R is an ordered list of n rules $\{r_1, r_2, \ldots, r_n\}$

- From this point on, assume first match model

No.	Proto.	Source IP	Source Port	Destination IP	Destination Port	Action
1	UDP	1.1.*	*	*	80	deny
2	TCP	2.*	*	1.*	90	accept
3	UDP	*	*	1.*	*	accept
4	TCP	2.*	*	1.*	20	accept
5	UDP	1.*	*	*	*	accept
6	*	*	*	*	*	deny

Accept Sets

- A policy default is executed when all other rules fail to match
- To reduce the policy size use a default rule:
  - Default 'deny'
  - Default 'accept'
- An **accept set** A is the set of all possible unique packets which a policy will accept
- A **deny set** D is the set of all possible unique packets which a policy will deny

  Definition A **comprehensive** policy R is one where $\bar{D} = A$

  Definition R and R' are **equivalent** if $A = A'$

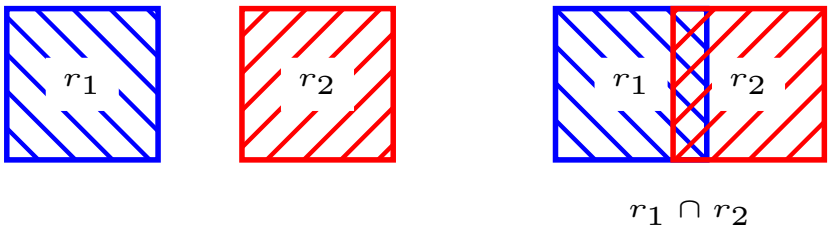
  Definition If R' is a modified R then **integrity** is maintained

Modeling Precedence

- Precedence modeled as a Directed Acyclical Graph (DAG)
  - Vertices are rules, edges are **precedence relationships**
  - Edge exists between r_i and r_j, if $i < j$ and the rules intersect
  - Rules **intersect** if their every tuple of their set intersection is non-empty

Definition The **intersection** of rule r_i and r_j, $(r_i \cap r_j)$

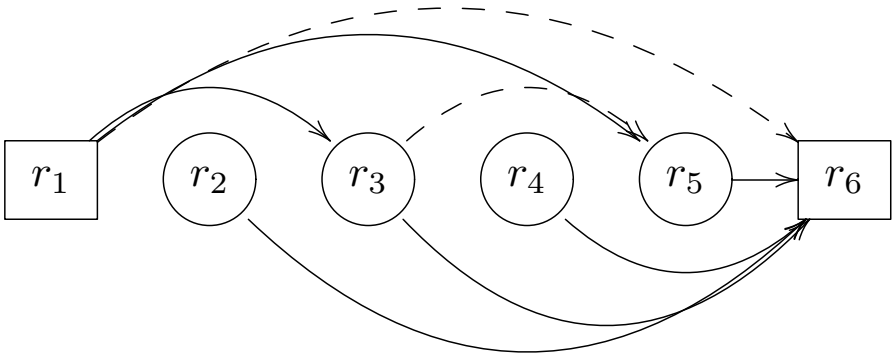
$$r_i \cap r_j = (r_i[l] \cap r_j[l]), \quad l = 1, \ldots, k$$



- Intersection describes the set of packets that match both rules
- If two rules intersect, then the order is significant

Precedence Relationships

No.	Proto.	Source IP	Source Port	Destination IP	Destination Port	Action
1	UDP	1.1.*	*	*	80	deny
2	TCP	2.*	*	1.*	90	accept
3	UDP	*	*	1.*	*	accept
4	TCP	2.*	*	1.*	20	accept
5	UDP	1.*	*	*	*	accept
6	*	*	*	*	*	deny



Discussion on Current Firewall Approaches

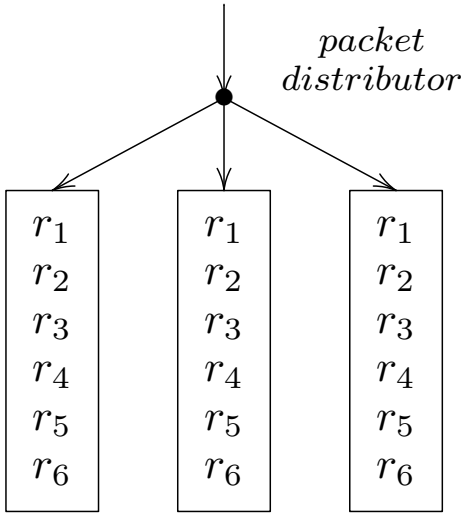
- Software Firewalls
  - User space vs Kernel space
  - NetFilter, SunScreen, IPFilter
  - Good development platform
- Hardware Firewalls
  - Edgeware Net Appliances
  - Cisco, Check Point
  - Closer to line speed
  - Dedicated logic, most use niche market devices
    - NPU – Network Processing Unit
    - ASIC – Application Specific Integrated Circuit
    - FPGA – Field Programmable Gate Array

Discussion on Current Firewall Approaches

- Ultimately Software approaches are bound to the limits of the OS:
  - Resource competitive environment
- Both solutions are limited by the hardware used
- Common solution is to buy bigger and faster machine
  - Non-modular
  - Not economically ideal
- Single points of entry can easily become overwhelmed in surges of traffic
  - Denial of Service
- Therefore there is a need for a scalable solution

Current Parallel Firewall Architectures

- An **array** of firewalls consists of m **firewall nodes**
- Each firewall node has a **local policy** to enforce
- **Definition** A system is **data parallel** (load-balancing) if:
  - Distributes packets evenly to all firewall nodes
  - Duplicates original policy to each firewall node ($R_i = R$)



Data Parallel, Overview

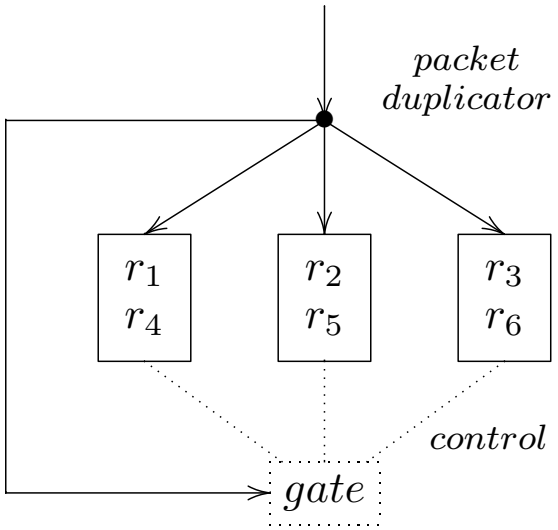
- Previously done by Benecke, then **Jeff Shirley**
- Packet distribution ensures no duplicates
- Maintains integrity since $A_i = A$
- Better throughput than traditional designs

Data Parallel, The Bad

- Does not allow for Quality of Service or state
- Benefit is related to load, when enough traffic exists to split
- Does not directly focus on reducing processing delay
  - Less transparent to users
- New parallel firewall architectures must solve these problems
  - To meet future demands
  - Increasing security threats

Function Parallel with Gate

- **Definition** A system is **function parallel** (with gate) if:
  - Duplicates packets to all firewall nodes
  - Distributes local policy R_i to each firewall node, where $\bigcup_{i=1}^{m} A_i = A$
  - A **gate** coordinates local policy results



Function Parallel, Why Gate?

- In this variation (**FPG**), precedence edges exist between firewall nodes
  - No firewall node can make a decision independently
- Incoming packets are duplicated to all firewalls and the gate
  - Multiple firewall nodes may find an accept match for the same packet if $A_i \cap A_j$, $i \neq j$
  - A gate node is needed to make a final decision

FPG, How the Gate Works

- Firewall nodes do not execute the associated action
  - Send decision as a **vote** to the gate
  - Vote consists of at least the rule number and action
    - **No match** is a valid response
    - Matches in state would have uniformally lower values
- The gate caches the packet until a decision can be made
- First match method is accomplished by executing the action of the vote with the lowest rule number

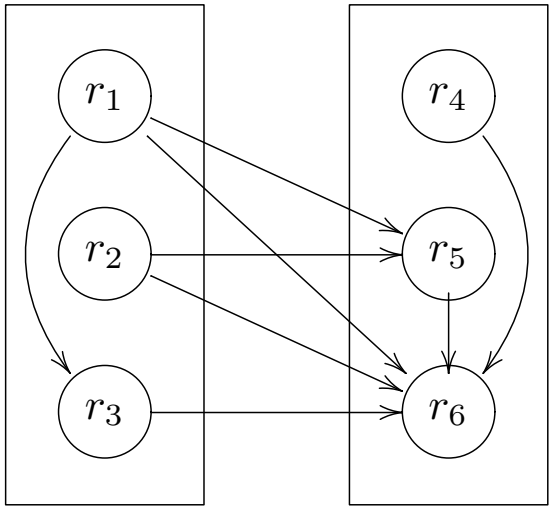
  How is last match done?

FPG, Integrity in Rule Distributions

- Local policies are distributed such that $\bigcup_{i=1}^{m} A_i = A$
- Gate resolves which rule is the appropriate final match, preserving rule precedence
- For example:
  - Put every rule on at least one machine
  - Never let the local policies contain shadowing
    - Local rule order always increases

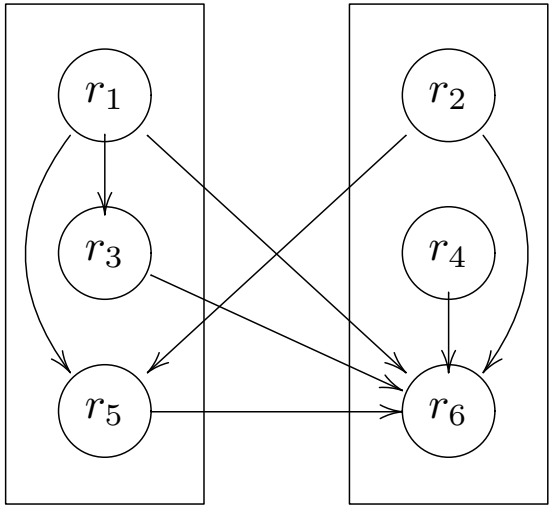
FPG, Example Rule Distributions

- **Vertical distribution**
  - Incrementally distribute rules
  - First $n\%m$ firewall nodes have $n/m + 1$ rules, rest have n/m



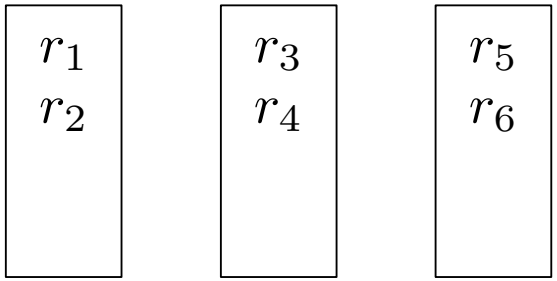
FPG, Example Rule Distributions

- **Horizontal distribution**
  - Incrementally distribute rules
  - Round robin



FPG, Failure

- If one firewall node fails. . . system would fail
- **Redundancy** is important
- Duplicating the entire system is inhibitive



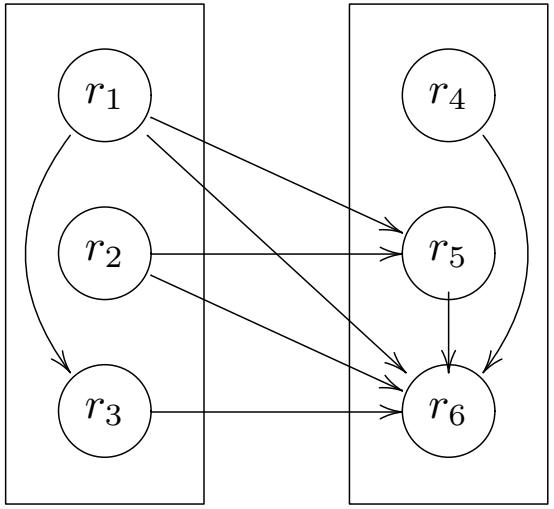
FPG, Redundancy

- Duplicate R_j by appending it to R_i where $i < j$, i.e. $i = j - 1$
- This requires an extra firewall node, so put append R_1 onto Rm
- Now $\bigcup_{i=1}^{m} A_i = A$ is still true
- Gate still prevent duplicates
- Performance could be increased with dynamic insertions on failure

r_1	r_3	r_1
r_2	r_4	r_2
r_3	r_5	r_5
r_4	r_6	r_6

FPG, Short-Circuit Evaluation

- Currently the system is as fast as the slowest firewall node in all cases
- Information from the DAG could be used to reduce the required votes



FPG, Short-Circuit

- If the gate machine can tell the firewall nodes to stop processing a packet:
  - Firewall nodes to move on to the next packet
  - Makes best time 1 rule and most cases less than worse case
    - Policy Default
    - Last rule on slowest machine
    - A rule with precedence from another machine
    - No precedence
  - Speeds up the processing time
- If the higher hit ratios were earlier in the vote array then you would really see performance increase

FPG, Pipelining the Process

- If the array processes packets asynchronously, then it increases work efficiency
- Would show performance benefits from short-circuit processing
  - firewall node could preemptively empty packets from a queue
  - implies firewall nodes track gate messages
- Throttle message might be necessary
- However, requires gate to track multiple packet decisions

FPG, Summary

- This method has distinct advantages over traditional and data parallel
  - Quality of Service
  - Stateful inspection
  - Reduced processing delay
- Disadvantages:
  - Is only limited by number of rules, which is generally not an issue
  - There is delay associated with the gate

Function Parallel with no Gate Design

- If the firewall nodes could be designed to act independently then the gate could be removed
- **Definition** A system is **function parallel**, and does not require a gate if:
  - Duplicates packets to all firewall nodes
  - Distributes a local policy R_i to each firewall node, where $\bigcup_{i=1}^{m} A_i = A$ and $\bigcap_{i=1}^{m} A_i = \emptyset$
- Incoming packets are duplicated to all firewalls and the gate
  - Since no accept sets intersect, only one firewall node will find an accepting match

FP, Integrity in Rule Distributions

- Local policies are distributed such that $\bigcup_{i=1}^{m} A_i = A$ and $\bigcap_{i=1}^{m} A_i = \emptyset$
- The last constraint guarantees no more than one firewall node will accept the same packet
- For example:
  - Put every rule on at least one machine
  - Never let the local policy DAGs contain shadowing
  - Divide the policy into non-intersecting local policies
- Consider the common case of a policy with no precedence edges and default deny

FP vs DP firewall, Theoretical Model

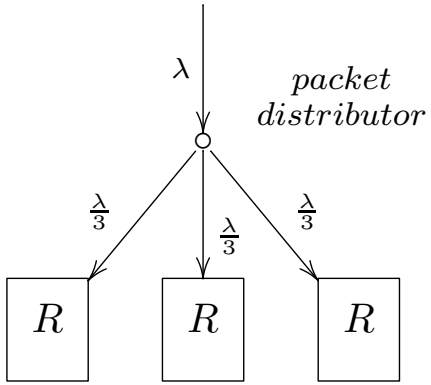
- Considered an open network of M/M/1 queues (Jackson Network)
  - A **queue** represents a **firewall node**
- Can be used to calculate an average of completely independent queues
- λ is the system **arrival rate**
- μ is processes per unit time, and $\frac{1}{\mu}$ is the **service time**
- Standard formula for delay of a cascading system is

$$E(T) = \sum_{i=1}^{q} \frac{1}{\mu_i - \lambda_i}$$

- But both DP and FP have a single layer of concurrent queues

FP vs DP firewall, Theoretical Model

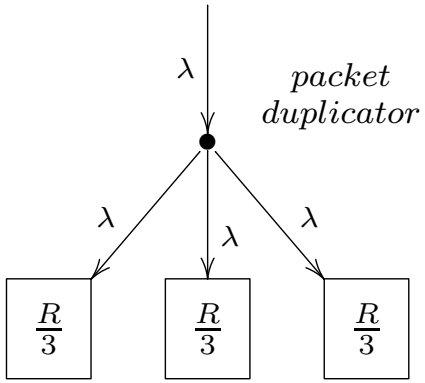
- Let x equal the rules processed per unit time
- For data parallel each firewall node
  - Arrival rate is $\frac{\lambda}{m}$
  - Processing time is $\frac{x}{n}$



$$E_d(T) = \frac{1}{\frac{x}{n} - \frac{\lambda}{m}}$$

FP vs DP firewall, Theoretical Model

- Let x equal the rules processed per unit time
- For function parallel each firewall node
  - Arrival rate is λ
  - Processing time is $\frac{x}{\frac{n}{m}} = \frac{m \cdot x}{n}$



$$E_f(T) = \frac{1}{\frac{m \cdot x}{n} - \lambda}$$

FP vs DP firewall, Theoretical Model

- Data parallel is then

$$E_d(T) = \frac{1}{\frac{x}{n} - \frac{\lambda}{m}}$$

- Function parallel is then

$$E_f(T) = \frac{1}{\frac{m \cdot x}{n} - \lambda}$$

- The reduction tells us the theoretical relation of delay (FP has $\frac{1}{m}^{th}$ the delay that DP firewall does):

$$\frac{E_f(T)}{E_d(T)} = \frac{1}{m}$$

FP, Summary

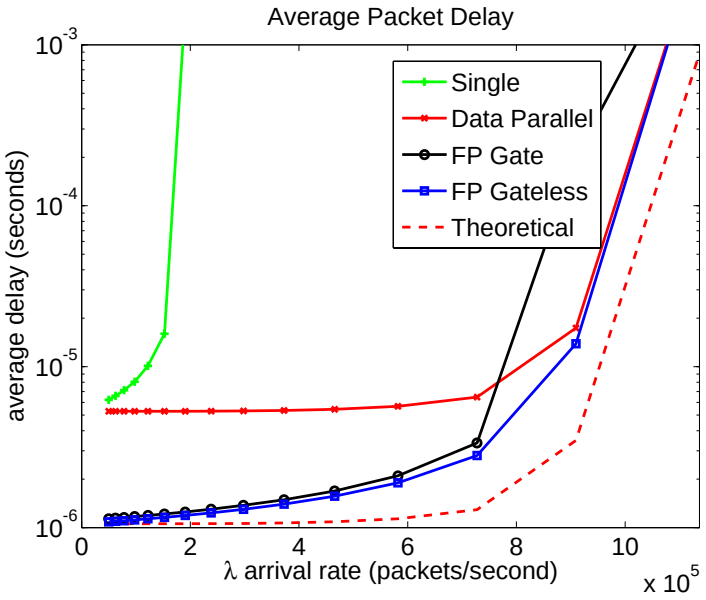
- This process has the same advantages as function parallel with gate
  - Quality of Service
  - Stateful inspection
  - Reduced processing delay
  - No additional gate delay
  - Compatible with legacy firewall systems
- Shares one disadvantage with the function parallel with gate:
  - Is only limited by number of rules, which is generally not an issue

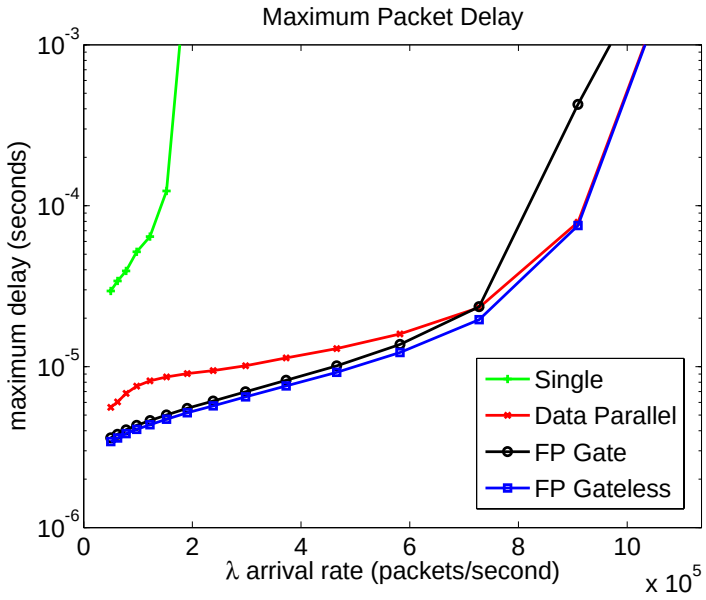
Parallel Firewall Simulation Results

- To compare all designs simulations were used
- Assumptions
  - Each firewall node could process 6×10^7 rules per second
  - Inter-arrival rate scheduled on **Poisson** distribution
  - Rule match probability according to **Zipf** distribution
  - No additional delay for DP firewall packet distributor
  - Costant gate delay for FPG
- Cases were ran to determine the performance of:
  - Increasing arrival rates
  - Increasing policy size
  - Increasing number of nodes

Delay vs Arrival Rate

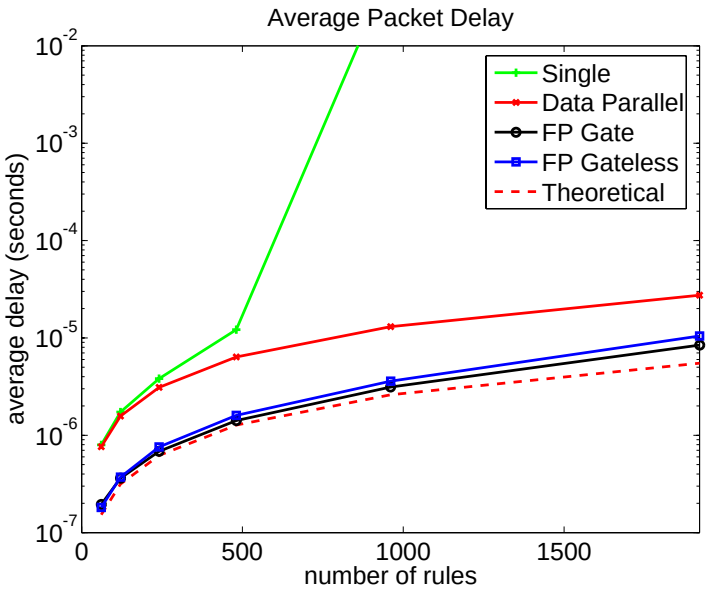
- Parallel systems consisted of 5 firewall nodes
- Policy size was 1024 rules
- Arrival rate was varied from 300 Mbps up to 6 Gbps

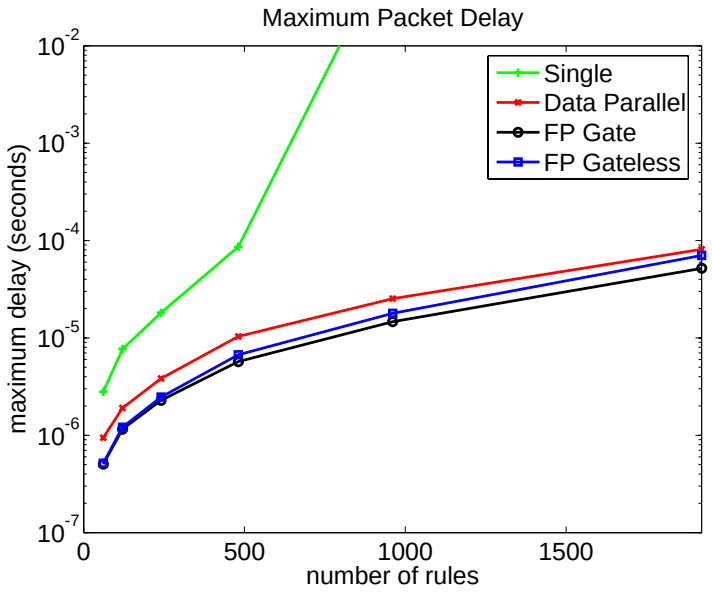




Delay vs Policy Size

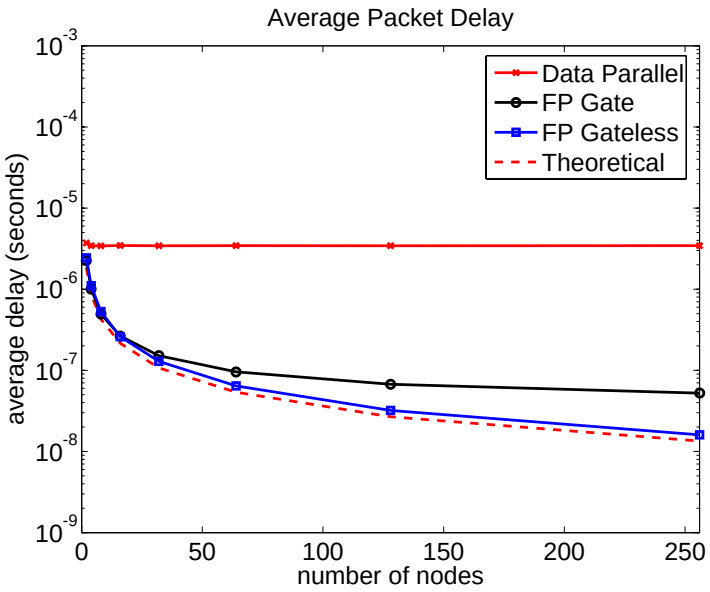
- Parallel systems consisted of 5 firewall nodes
- Arrival rate was established at 650 Mbps
- Policy size was incremented from 2 to 2048

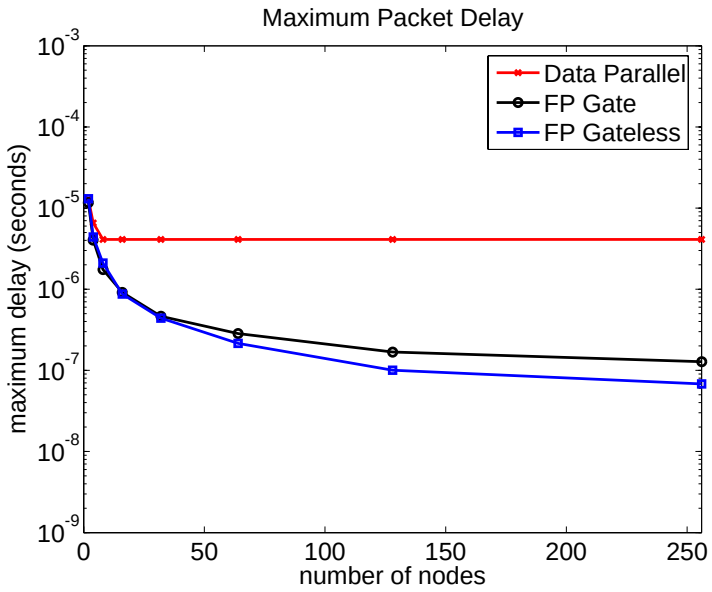




Delay vs Number of Firewall nodes

- Arrival rate was established at 650 Mbps
- Policy size was 1024 rules
- Parallel systems varied number of firewall nodes from 2 to 256





Summary of Simulations

- Illustrates advantage of parallelism
- Reducing **processing time** is more advantageous than reducing **arriving traffic load**
- Removing the gate delay helps function parallel approach theoretical rates

How to Roll Your Own

- System can be divided into components
  - Firewall nodes – Linux PC running iptables
  - Packet Duplicator
    - 10/100 Mbps use a hub
    - Gigabit requires a tap (usually used for IDS)
  - Control Plane
    - Needed to contact firewall nodes for management
    - Uses separate subnet for security

How to Roll Your Own

- Combining components
  - Firewall nodes
    - Duplicate IPs and MACs in stealth mode
    - One IP/MAC per incoming interface
    - Enable promiscuous mode and disable ARPs
    - Disable ICMP requests
    - Consider enabling one firewall node to allow ARP and ping
- Network topology given on board

Conclusions

- It is important that a firewall acts transparently to users
- Unfortunately, firewalls quickly become bottlenecks
- Particularly in **High Speed Networks**
- Improving implementations and hardware is not as scalable as needed
- Enter **Parallel firewalls**
- **Data parallel** does not address processing delay
- **Function parallel with gate** is flexible, but has the added gate delay
- **Function parallel** with no gate solves scalable processing delay issues

Great Wall Systems

- Recently founded through WFU OTAM
- Basis is two patents created through research from DOE grant
- Dedicated to High Speed Networking Devices
- Located at 111 Chestnut Street, Victoria Hall, Winston-Salem, NC

That's All...

- Thank you for your time