

# High-Speed Packet Filtering Utilizing Stream Processors

Richard J. Hummel<sup>1</sup> and Errin W. Fulp<sup>1,2</sup>

<sup>1</sup>Wake Forest University  
Department of Computer Science, NC USA  
Winston-Salem, NC USA

<sup>2</sup>GreatWall Systems  
Winston-Salem

SPIE Conference Security and Defense  
April 15, 2009

# Need for High-Speed Firewalls

- Firewalls remain the forefront defense for computer systems
  - Inspect packets sent between networks
  - Provide access control, auditing, and traffic control
  - Actions performed on packets specified by security policy
- Similar to a router, a firewall is a single dedicated machine
  - Inspecting packets introduces delay (latency)
  - As high-speed networks become more prevalent, delays will become more significant
  - We are interested in reducing the latency across the system

# Firewall Policy and Rules

No.	Proto.	Source		Destination		Action	Prob.
		IP	Port	IP	Port		
1	UDP	190.1.1.*	*	*	80	deny	0.05
2	UDP	210.1.*	*	*	90	accept	0.10
3	TCP	180.*	*	180.*	90	accept	0.15
4	TCP	210.*	*	220.*	80	accept	0.20
5	UDP	190.*	*	*	*	accept	0.20
6	*	*	*	*	*	deny	0.30

- Firewall rule consists of a 5-tuple and an action (IP networks)

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

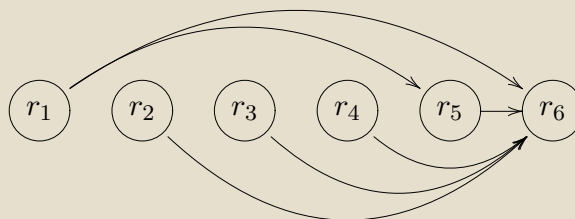
- Security policy is a ordered list of rules

$$R = \{r_1, r_2, \dots, r_n\}$$

- Packets sequentially compared with rules until *first match*

# Policy Attributes

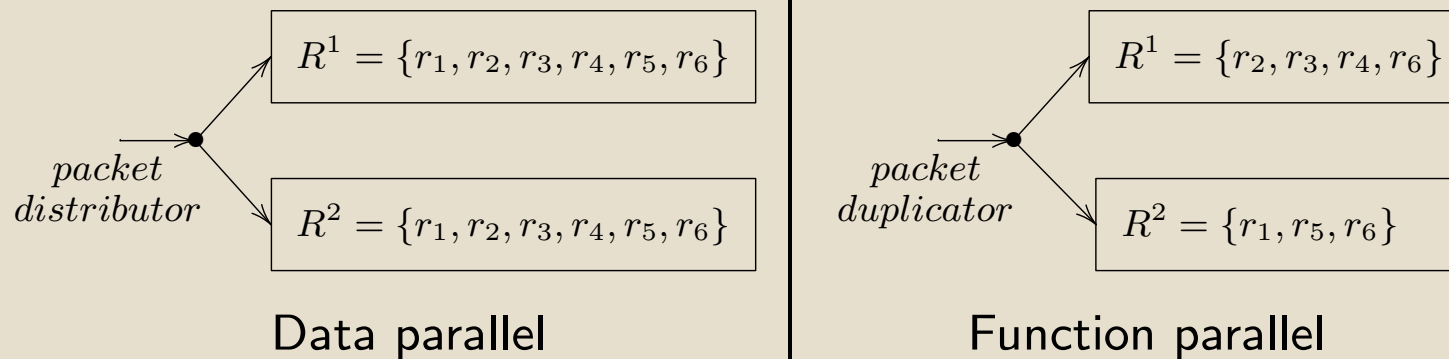
- Policy rules may intersect (*a packet can match more than one rule*)



- Every policy  $R$  has the following sets
  - Accept set  $A$ , deny set  $D$ , and no-match set  $N$
- A policy is **comprehensive** if  $A \cup D = P$  or  $N = \emptyset$
- Consider two policies  $R$  and  $R'$ 
  - Policies **equivalent** if comprehensive and  $A = A'$
  - **Integrity** is maintained if  $R$  is replaced with  $R'$
  - *Integrity applies to policies and implementation*

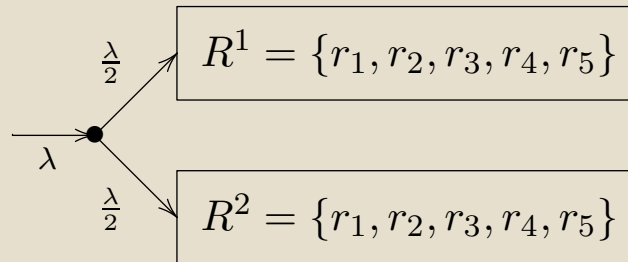
# Parallel Firewall Designs

- Parallel firewalls can be used to reduce packet processing time
- Given an array of  $m$  machines, consider two parallel designs
  - **Data parallel**, distribute packets
  - **Function parallel**, distribute rules *with restrictions*



- Make certain **integrity** is maintained for each design

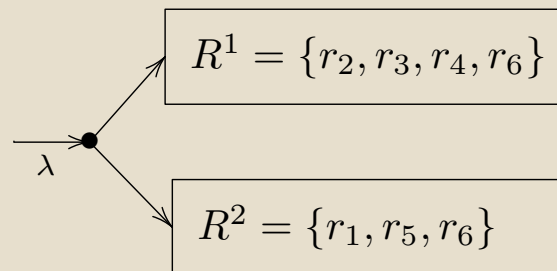
# Data Parallel



- Multiple firewalls connected in parallel [Ben99]
  - Packets distributed across the firewalls ( $\frac{\lambda}{m}$ ), where  $\lambda$  is the arrival rate and  $m$  is the number of firewalls
  - Each firewall implements  $R^i = R, \quad i = 1, \dots, m$
- Maintains integrity since  $A^i = A, \quad i = 1, \dots, m$
- Scales (does not depend on rule set) and robust
- Difficult to provide service differentiation and state information

# Function Parallel

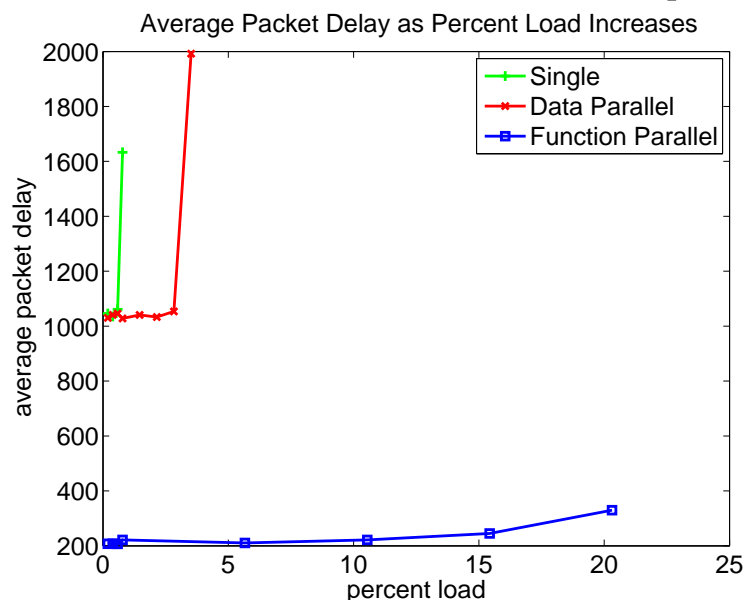
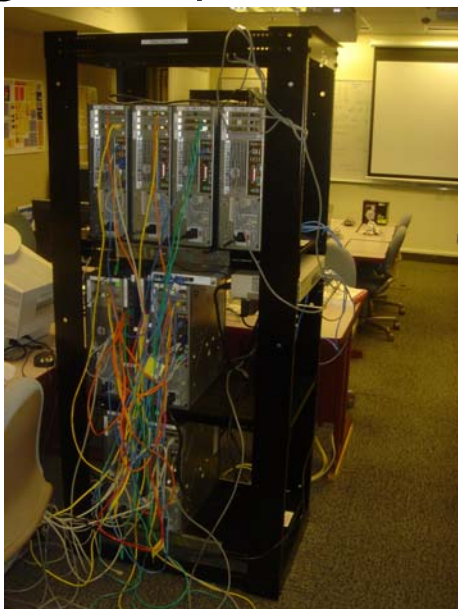
- An array of  $m$  firewalls connected in parallel [FF06]
  - Rules are distributed and packets are duplicated



- A packet is processed by each firewall in parallel
  - Rules must be distributed to maintain integrity
- Has several advantages over data parallel design
  - Scalable and maintains state information
  - Faster under low traffic load, *given proper rule distribution* [HFW08]

# Firewall Implementations

- Original implementation used multiple BSD firewalls [Ful06]

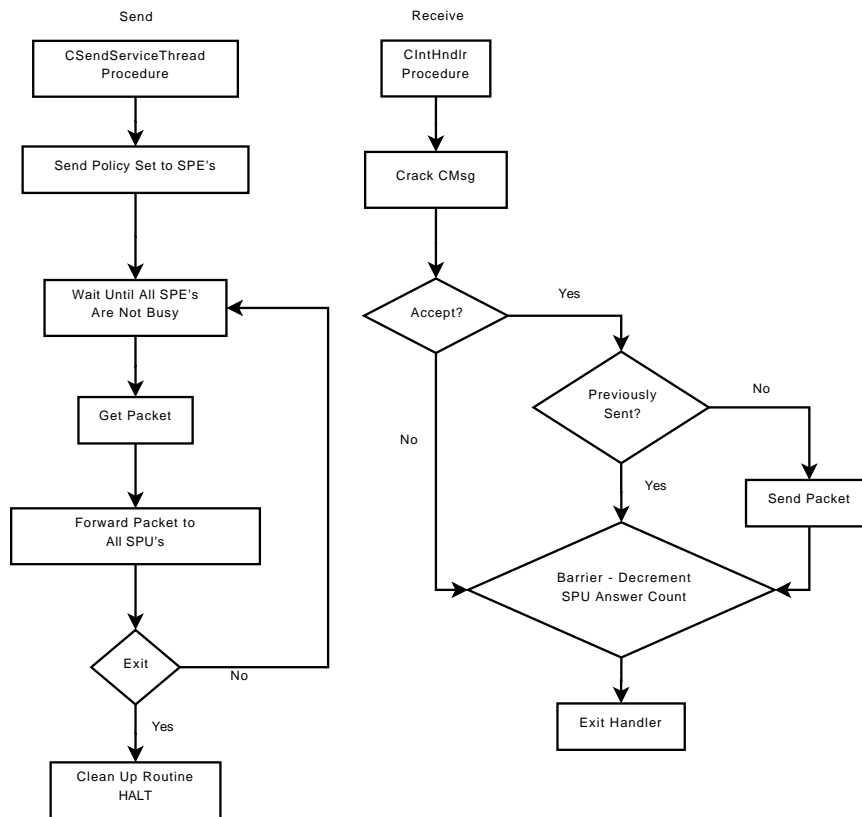


- Results were comparatively good (*of the designs*)
  - Function parallel better than data parallel
  - Overall speed was limited to 20 Mbps
- *Multi-core system should provide significant improvements*



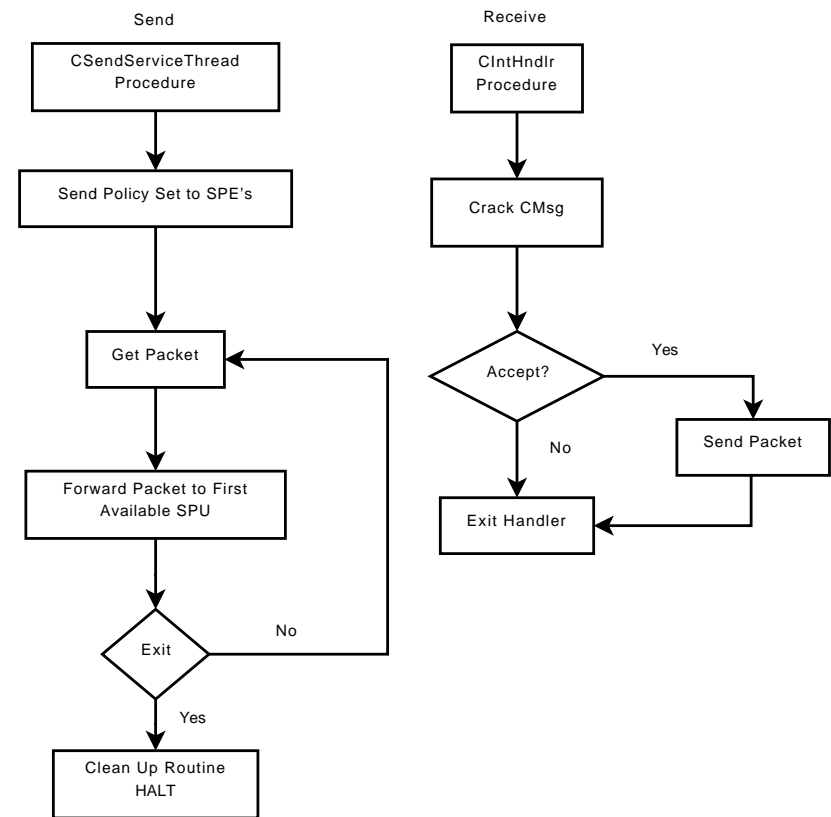
# CBE Packet Processing

## Function Parallel



Packets processed in *lock-step*

## Data Parallel



Packets distributed

# Performance Evaluation

- CBE-based packet filter using Sony PlayStation3 (PS3)
  - 3.2 GHz CBE, 256 MB RAM, and 1 Gbps NIC
  - One PPU (PowerPC) and six SPEs
  - Fedora Core 6, kernel 2.6.23
- Two tests performed using RFC 3511 testing specification
  - Internal packet generation
  - External packet generation

# Internal Packet Generation

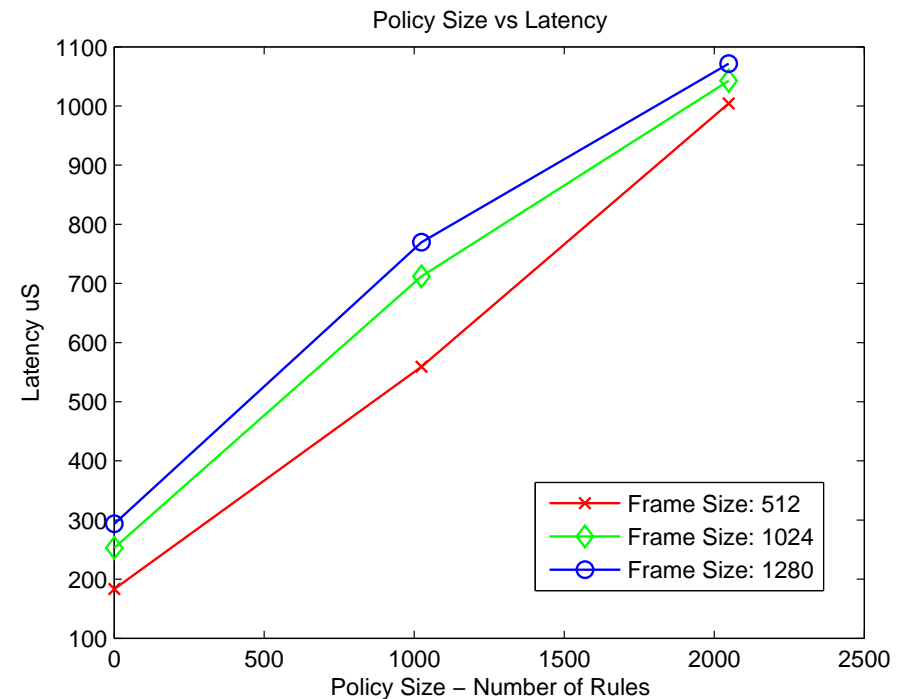
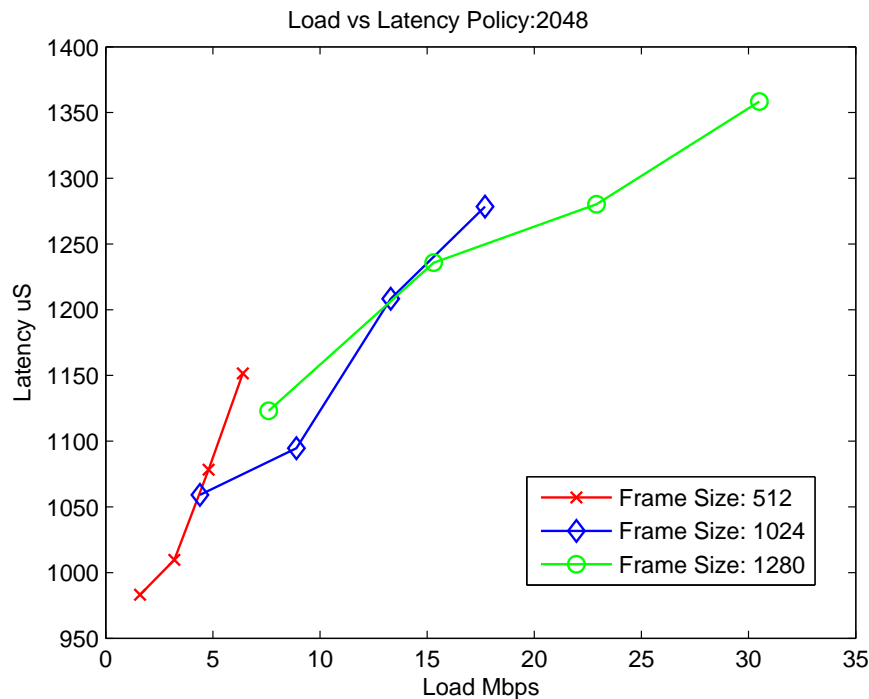
- Determine the processing ability of the CBE
  - Packets generated within the CBE (*no networking subsystem*)
  - Different policy sizes and 1500 B packets

Policy Size	Data Parallel		Function Parallel	
	Max Throughput (Mbps)	Min Latency ( $\mu$ -sec)	Max Throughput (Mbps)	Min Latency ( $\mu$ -sec)
0	939.5	81.5	55.2	1000
1024	249.1	422.2	57.6	1577
2048	131.6	771.4	53.2	2420
4096	73.9	1472	33.8	3487

- Function parallel has lower throughput and higher latency
  - Lock-step design is not efficient, only one PPU...
- Data parallel performance comparable to other related research

# External Packet Generation

- Measure performance of actual packet filtering (*data parallel only*)
  - Packets generated using SmartBits 200 (100 Mbps)







- System performed better with larger frames and smaller policies
  - Overall performance limited by libpcap **not** the processor

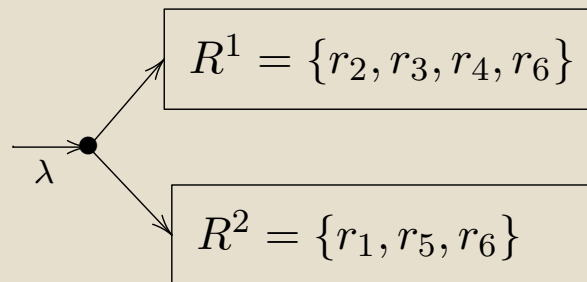
# Future Work

- Next generation high-speed security systems
  - Maintain QoS, provide differentiation, and scalable
  - Parallel firewalls can improve performance
- Using CBE must avoid libpcap for reading and writing packets
  - Used PF\_RING with limited success
  - BSD provide better model for packet processing
- Tileria 64 network processing card
  - Designed for packet processing, consists of 64 cores
  - Tested to 10 Gbps, should scale over 40 Gbps
- *Deep packet inspection?*

# Bibliography

-  Carsten Benecke.  
A parallel packet screen for high speed networks.  
In Proceedings of the 15th Annual Computer Security Applications Conference, 1999.
-  Errin W. Fulp and Ryan J. Farley.  
A function-parallel architecture for high-speed firewalls.  
In Proceedings of the IEEE International Conference on Communications, 2006.
-  Errin W. Fulp.  
An independent function-parallel firewall architecture for high-speed networks (short paper).  
In Proceedings of the International Conference on Information and Communications Security, 2006.
-  Michael R. Horvath, Errin W. Fulp, and Patrick S. Wheeler.  
Policy distribution methods for function parallel firewalls.  
In Proceedings of the IEEE International Conference on Computer Communications and Networks (ICCCN), 2008.

# Function Parallel Firewall Policy Integrity



- Integrity maintained if rules distributed using accept sets

## Theorem

*A function parallel array of  $m$  firewalls enforcing a comprehensive policy  $R$  maintains integrity if policy rules are distributed such that: each local policy is comprehensive,  $\bigcup_{j=1}^m A^j = A$ , and  $\bigcap_{j=1}^m A^j = \emptyset$ .*