A Case for Spraying Packets in Software Middleboxes

Hugo Sadok, Miguel Elias M. Campista, Luís Henrique M. K. Costa
Middleboxes
Middleboxes
Software Middleboxes

Purpose-built hardware
Software Middleboxes

Purpose-built hardware  Commodity servers
Software Middlebox Architecture

Queue Selection

NIC (Rx)

L3 Cache

Core 0
Core 1
Core 2
Core 3

CPU

Queue Arbiter

NIC (Tx)
Software Middlebox Architecture

NIC selects core using a hash of the 5-tuple (RSS: Receive-Side Scaling)
Core processes the packet
Software Middlebox Architecture

Queue Selection

NIC (Rx)

L3 Cache

Core 0
Core 1
Core 2
Core 3

CPU

NIC (Tx)

Queue Arbiter

NIC transmits the packet

Queue Selection

NIC (Rx)
Software Middlebox Architecture

Queue Selection

NIC (Rx)

L3 Cache

Core 0
Core 1
Core 2
Core 3

CPU

Queue Arbiter

NIC (Tx)
Why hash flows to cores?
Software Middlebox Architecture

Why hash flows to cores?
1. Avoids packet reordering
2. Facilitates flow state handling
Problems with Hashing Flows to Cores

Inefficiency
Problems with Hashing Flows to Cores

Queue Selection

NIC (Rx)

L3 Cache

Queue Arbiter

CPU

NIC (Tx)

Inefficiency

Unfairness
Problems with Hashing Flows to Cores

Inefficiency

Unfairness
Problems with Hashing Flows to Cores

Queue Selection

NIC (Rx)

Core 0

Core 1

Core 2

Core 3

L3 Cache

Queue Arbiter

NIC (Tx)

CPU

Inefficiency

Unfairness
Problems with Hashing Flows to Cores
Problems with Hashing Flows to Cores

No matter how you do it, mapping flows to cores is inefficient and unfair.
Analogous Problem in Datacenters

- Multiple paths
- Low and similar latencies
Analogous Problem in Datacenters

- Multiple paths
- Low and similar latencies

RPS [INFOCOM ’13]
pFabric [SIGCOMM ’13]
NDP [SIGCOMM ’17]
Hermes [SIGCOMM ’17]
Analogous Problem in Datacenters

- Multiple paths
- Low and similar latencies

Packet Spraying

- RPS [INFOCOM ’13]
- pFabric [SIGCOMM ’13]
- NDP [SIGCOMM ’17]
- Hermes [SIGCOMM ’17]
Can software middleboxes also benefit from load balancing packets at a finer granularity?
Can software middleboxes also benefit from load balancing packets at a finer granularity?

Sprayer
Can software middleboxes also benefit from load balancing packets at a finer granularity?

Sprayer

1. Efficiently handle flow state
2. Spray packets using existing NICs
How to efficiently handle flow state?
How to efficiently handle flow state?

NFs have two types of state: local and global
How to efficiently handle flow state?

When all packets from the same flow go to the same core, flow state is *partitionable*.
How to efficiently handle flow state?

If we spray packets and let every core update the state for a given flow, we lose this property…
How to efficiently handle flow state?

However, we get similar benefits if we provide writing partition
How to efficiently handle flow state?

However, we get similar benefits if we provide **writing partition**
How to efficiently handle flow state?

However, we get similar benefits if we provide writing partition
How to efficiently handle flow state?

However, we get similar benefits if we provide writing partition.
How to achieve writing partition?
How to achieve writing partition?

Many NFs only need to change flow state when TCP connections start or finish.

e.g., NAT, firewall, load balancer, traffic monitor
How to achieve writing partition?

Many NFs only need to change flow state when TCP connections start or finish

*e.g.*, NAT, firewall, load balancer, traffic monitor

Ensure that packets at the beginning or end of the same TCP connection go to the same core
Sprayer

1. Efficiently handle flow state ✔
2. Spray packets using existing NICs
How to spray packets using existing NICs?
How to spray packets using existing NICs?

Two ways to direct packets to cores
How to spray packets using existing NICs?

Two ways to direct packets to cores

**RSS:** use hash of the 5-tuple
How to spray packets using existing NICs?

Two ways to direct packets to cores

**RSS**: use hash of the 5-tuple

**Flow Director**: match arbitrary header field
How to spray packets using existing NICs?

Two ways to direct packets to cores

**RSS:** use hash of the 5-tuple

**Flow Director:** match arbitrary header field
How to spray packets using existing NICs?

**TCP Header**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source port</td>
<td>Source port of the packet</td>
</tr>
<tr>
<td>destination port</td>
<td>Destination port of the packet</td>
</tr>
<tr>
<td>sequence number</td>
<td>Sequence number of the packet</td>
</tr>
<tr>
<td>acknowledgement number (ACK)</td>
<td>Acknowledgement number</td>
</tr>
<tr>
<td>offset</td>
<td>Offset</td>
</tr>
<tr>
<td>reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>flags</td>
<td>Flags</td>
</tr>
<tr>
<td>window size</td>
<td>Window size</td>
</tr>
<tr>
<td>checksum</td>
<td>Checksum</td>
</tr>
<tr>
<td>urgent pointer</td>
<td>Urgent pointer</td>
</tr>
</tbody>
</table>
How to spray packets using existing NICs?

<table>
<thead>
<tr>
<th>TCP Header</th>
<th>source port</th>
<th>destination port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sequence number</td>
<td></td>
</tr>
<tr>
<td></td>
<td>acknowledgement number (ACK)</td>
<td></td>
</tr>
<tr>
<td>offset</td>
<td>reserved</td>
<td>flags</td>
</tr>
<tr>
<td>checksum</td>
<td></td>
<td>urgent pointer</td>
</tr>
</tbody>
</table>
1. Efficiently handle flow state
2. Spray packets using existing NICs
Preliminary Evaluation
Will packet reordering have a significant impact on TCP?
Preliminary Evaluation

- Will packet reordering have a significant impact on TCP?
- How much improvement do we get from Sprayer?
Preliminary Evaluation

Throughput

Latency

Fairness
Preliminary Evaluation

- Consistent throughput, regardless of the number of concurrent flows
Preliminary Evaluation

- Consistent throughput, regardless of the number of concurrent flows
- Significant improvement for small number of flows
Preliminary Evaluation

- Consistent throughput, regardless of the number of concurrent flows
- Significant improvement for small number of flows
- Packet reordering is not enough to significantly harm TCP
## Preliminary Evaluation

### Throughput
- Consistent throughput, regardless of the number of concurrent flows
- Significant improvement for small number of flows
- Packet reordering is not enough to significantly harm TCP

### Latency
- Sprayer processes packets from the same flow in parallel, reducing the latency

### Fairness
Preliminary Evaluation

**Throughput**
- Consistent throughput, regardless of the number of concurrent flows
- Significant improvement for small number of flows
- Packet reordering is not enough to significantly harm TCP

**Latency**
- Sprayer processes packets from the same flow in parallel, reducing the latency

**Fairness**
- Sprayer achieves nearly perfect fairness
Preliminary Evaluation

**Throughput**
- Consistent throughput, regardless of the number of concurrent flows
- Significant improvement for small number of flows
- Packet reordering is not enough to significantly harm TCP

**Latency**
- Sprayer processes packets from the same flow in parallel, reducing the latency

**Fairness**
- Sprayer achieves nearly perfect fairness
- RSS suffers from hash collisions
Takeaways
Takeaways

- Matching flows to cores causes inefficiency and unfairness
Takeaways

- Matching flows to cores causes inefficiency and unfairness
- We need to load balance packets at a finer granularity
Takeaways

- Matching flows to cores causes inefficiency and unfairness
- We need to load balance packets at a finer granularity
- Sprayer takes the first step by:
  - Ensuring that flow states are handled efficiently
  - Working with commodity NICs
Open Questions
Open Questions

- What about other types of NFs?
Open Questions

- What about other types of NFs?
- Can Sprayer benefit from programmable NICs?
Open Questions

- What about other types of NFs?
- Can Sprayer benefit from programmable NICs?
- What about other transport protocols?