Flexplane: An Experimentation Platform for Resource Management in Datacenters

Amy Ousterhout, Jonathan Perry, Hari Balakrishnan, Petr Lapukhov
Datacenter Networks

• Applications have diverse requirements
• Dozens of new resource management schemes
  – Low latency: DCTCP
  – Min FCT: PDQ, RCP, pFabric, PERC
  – Deadlines: D³, D²TCP
• Difficult to experiment with schemes in real networks
  – Requires changes to hardware routers
Experimentation with Resource Management

• Experimentation in real networks
  – Software routers - limited **throughput**
  – Programmable hardware - limited **flexibility**
Experimentation with Resource Management

• Experimentation in simulation (e.g., ns, OMNeT++)
  – Does not accurately model real network stacks, NICs, and distributed applications
  – Does not run in real time

No existing approach to experimentation provides accuracy, flexibility, and high throughput
Our Contributions

• Key idea: whole-network emulation
• Flexplane: a platform for faithful experimentation with resource management schemes
  – Accurate – predicts behavior of hardware
  – Flexible – express schemes in C++
  – High throughput – 761 Gbits/s
Approach: Whole-Network Emulation

Real Network

Emulated Network

class MyScheduler {...}
class MyAQM {...}
Abstract Packets

• Resource management schemes are *data-independent*

• Concise representation of one MTU
  – Source, destination, flow, ID
  – Custom per-scheme fields
Emulator

• Real-time network simulator
• Faster than standard network simulators
  – Time divided into abstract-packet-sized timeslots
  – Omits endpoint software
Accuracy

• Goal: predict behavior of a hardware network
• Hardware latency:
• Added latency of Flexplane:
  – RTT to emulator
  – Unloaded delay
  – Queuing delay
  in real network
Flexplane API

- Decouples schemes from framework

<table>
<thead>
<tr>
<th>Emulator</th>
<th>int route(AbstractPkt *pkt)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>int classify(AbstractPkt *pkt, int port)</td>
</tr>
<tr>
<td></td>
<td>enqueue(AbstractPkt *pkt, int port, int queue)</td>
</tr>
<tr>
<td></td>
<td>AbstractPkt *schedule(int output_port)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Endpoints</th>
<th>prepare_request(sk_buff *skb, char *request_data)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>prepare_to_send(sk_buff *skb, char *allocation_data)</td>
</tr>
</tbody>
</table>

Endpoints prepare_request(sk_buff *skb, char *request_data)
prepare_to_send(sk_buff *skb, char *allocation_data)

Incoming packets → route → classify → enqueue → schedule → outgoing packets
Multicore Emulator Architecture

- Pin network components (routers, endpoints) to cores
- Communication via FIFO queues
- Router state not shared across cores
Implementation

• Emulator uses Intel DPDK for low-latency NIC access
• Endpoints run a Linux qdisc
Evaluation

• Accuracy
• Utility
• Emulator throughput
Flexplane is Accurate

- Bulk TCP: 5 senders, 1 receiver
- Throughput 9.2-9.3 Gbits/s vs. 9.4 Gbits/s in hardware
- Similar queue occupancies

<table>
<thead>
<tr>
<th>Median Queue Occupancies (MTUs)</th>
<th>Hardware</th>
<th>Flexplane</th>
</tr>
</thead>
<tbody>
<tr>
<td>DropTail</td>
<td>931</td>
<td>837</td>
</tr>
<tr>
<td>RED</td>
<td>138</td>
<td>104</td>
</tr>
<tr>
<td>DCTCP</td>
<td>61</td>
<td>51</td>
</tr>
</tbody>
</table>
Flexplane is Accurate

- RPC web search workload

- Accurate to within 2-14% for loads up to 60%

- Observe behavior not visible in simulations
Flexplane is Easy to Use

- Implemented several schemes in dozens of lines of code

<table>
<thead>
<tr>
<th>scheme</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>drop tail queue manager</td>
<td>39</td>
</tr>
<tr>
<td>RED queue manager</td>
<td>125</td>
</tr>
<tr>
<td>DCTCP queue manager</td>
<td>43</td>
</tr>
<tr>
<td>priority queueing scheduler</td>
<td>29</td>
</tr>
<tr>
<td>round robin scheduler</td>
<td>40</td>
</tr>
<tr>
<td>HULL scheduler</td>
<td>60</td>
</tr>
<tr>
<td>pFabric QM, queues, scheduler</td>
<td>251</td>
</tr>
</tbody>
</table>
Flexplane Enables Experimentation

- Evaluating trade-offs between resource management schemes
Flexplane Enables Experimentation

- Experiment with real distributed applications such as Spark

<table>
<thead>
<tr>
<th>% Change in Completion Time</th>
<th>DCTCP</th>
<th>HULL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coordinate descent</td>
<td>+4.4%</td>
<td>+29.4%</td>
</tr>
<tr>
<td>Sort</td>
<td>-4.8%</td>
<td>-2.6%</td>
</tr>
</tbody>
</table>

- Performance depends on network and CPU
Emulator Throughput

• Emulator provides 761 Gbits/s of aggregate throughput with 10 total cores

• 81x as much throughput per clock cycle as RouteBricks
Flexplane: an Experimentation Platform

- Whole-network emulation
- Flexplane: a platform for faithful experimentation with resource management schemes
  - Accuracy, flexibility, and high throughput

https://github.com/aousterh/flexplane