A Linux Kernel Implementation of the Homa Transport Protocol

John Ousterhout
Overview

- Homa: a new transport protocol for datacenters
  - Low tail latency for small messages, even under high load
  - Previously evaluated with simulations, RAMCloud implementation

- Personal mission: replace TCP!

- Personal project: Linux kernel implementation

- Results: Homa tail latency 10-100x better than TCP

- Very difficult to implement high-performance protocols in Linux

- Transport protocols no longer make sense in software: must move to NIC
Homa Basics

- Receiver-driven flow control limits queue lengths
- Use switch priority queues to implement SRPT
- Receiver determines priorities dynamically
More on Homa

- **Low latency for short messages, even under high network load**
  - Uses switch priority queues to implement SRPT
  - Long messages have lower latency also (run to completion)

- **Prevents incast congestion**
  - Receiver-driven flow control limits queue lengths
  - Slight overcommitment/buffering to maintain link utilization
  - Priorities preserve SRPT even with buffering

- **Other features**
  - Message-oriented (not streaming)
  - Connectionless
    - State only for active RPCs
    - One socket per process, not per connection
Homa Project Evolution

● Behnam Montazeri’s dissertation project

● 2 evaluations:
  ▪ Simulations (Behnam Montazeri)
  ▪ RAMCloud user-space implementation (Yilong Li)

● Great results:
  ▪ Dominates all comparison protocols across many workloads
  ▪ 10-100x lower tail latency than TCP
  ▪ P99 latency under load typically within 2-3x of unloaded latency

● Next steps:
  ▪ How to gain wide adoption in the datacenter?
  ▪ Build Linux kernel implementation
  ▪ Eventually, build into NIC
HomaModule

- Dynamically loadable Linux kernel module
- Open source: `git@github.com:PlatformLab/HomaModule.git`
- 10,000 lines C code (heavily commented)
- Status:
  - Functionally complete
  - Still analyzing and tuning performance
  - Fairly stable/robust (but still finding bugs)
- Preliminary performance takeaways:
  - Significantly better than TCP
  - Yet still pathetic
## Basic Performance

<table>
<thead>
<tr>
<th></th>
<th>Homa</th>
<th>TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTT latency (unloaded, 100B messages)</td>
<td>18 µs</td>
<td>21 µs</td>
</tr>
<tr>
<td>Client RPC throughput (small messages)</td>
<td>1.7 M/s</td>
<td>1.2 M/s</td>
</tr>
<tr>
<td>Server RPC throughput (small messages)</td>
<td>1.8 M/s</td>
<td>1.4 M/s</td>
</tr>
<tr>
<td>Client large-message throughput</td>
<td>2.7 MB/s</td>
<td>2.7 MB/s</td>
</tr>
<tr>
<td>Server large-message throughput</td>
<td>2.7 MB/s</td>
<td>2.9 MB/s</td>
</tr>
</tbody>
</table>

CloudLab xl170 cluster:  
10 nodes, Intel E5-2640v4@2.4 GHz, 20 cores  
25 Gbps network, Mellanox ConnectX-4 NIC
Slowdown vs. TCP

Somewhat heavy-tailed workload

80% of max sustainable goodput

W4 10 nodes, 2.20 GB/s

TCP P99
Homa P99
TCP P50
Homa P50

SRPT effect

actual latency
unloaded Homa latency

x-axis: CDF of message length

Slowness
Slowdown, All Workloads

- W1 10 nodes, 0.18 GB/s
- W2 10 nodes, 0.30 GB/s
- W3 10 nodes, 1.80 GB/s
- W4 10 nodes, 2.20 GB/s
- W5 10 nodes, 2.20 GB/s
A Few Priority Levels is Enough

Homa P99 still better than TCP P50 even with only 1 priority level

Short-message workloads don’t need priorities (software limited)
Next Steps: Industrial Trial

- Looking for company interested in testing Homa internally
  - How does Homa compare to your current state of the art?
  - Do real applications see benefits?

- Prefer to be personally involved
  - Help with integration
  - Run tests
  - Fix bugs and performance issues
Challenges

- **Very difficult to implement a high-performance transport protocol in software**
  - Networks getting faster
  - CPUs not getting faster
  - Even harder in an OS like Linux (many layers)

- **Latency**

- **RPC throughput**

- **Utilizing fast networks**
  - Load balancing
  - Packet aggregation
Latency

- **Unloaded latency too high:**
  - TCP: 21 µs
  - Homa: 18 µs
  - Snap: 10 µs
  - RAMCloud: 4.7 µs

- **P99 slowdown also too high:**
  - Homa: 10x
  - RAMCloud: 2-3x
  - Simulations: 2-3x

- No OS implementation can get close to hardware potential (3 – 3.5 µs)
Latency (100B Messages)

<table>
<thead>
<tr>
<th>Device Driver</th>
<th>IP/Net</th>
<th>Homa Protocol</th>
<th>Homa Syscall</th>
<th>User Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 µs</td>
<td>0.6 µs</td>
<td>2.4 µs</td>
<td>2.6 µs</td>
<td>3.7 µs</td>
</tr>
</tbody>
</table>

| Interrupt       | NAPI Thread | SoftIRQ Thread | Application Thread |

<table>
<thead>
<tr>
<th>Total</th>
<th>Homa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive</td>
<td>4.2 µs</td>
</tr>
<tr>
<td>Send</td>
<td>1.1 µs</td>
</tr>
</tbody>
</table>
Throughput

- **High software overheads:**
  - Handling short RPCs on server side:
    - Homa: 4 cores/ 1M RPCs/
    - RAMCloud: 1 core/ 1M RPCs/sec
    - MICA: 0.2 core/ 1M RPCs/sec
  - Utilizing network bandwidth with large messages:
    - Homa: 0.12 core/Gbps (24 cores for 100 Gbps)
    - Snap: 0.06 – 0.09 core/Gbps (12-18 cores for 100 Gbps) (but only with 5000B MTU)

- **Must balance load across many cores**
  - Hard to do well

- **Must aggregate packets for stack traversal**
  - Linux: GSO, GRO
  - Complex mechanism (Homa impersonates TCP packets)
  - Increases receive latency
Throughput

- High overheads for short messages; server-side:
  - Homa: 0.25 M RPCs/sec/core
  - RAMCloud: 1 M RPCs/sec/core
  - MICA: 5 M RPCs/sec/core

- Expensive to utilize full network bandwidth
  - Homa: 0.12 core/Gbps (24 cores for 100 Gbps)
  - Snap: 0.06 – 0.09 core/Gbps (12-18 cores for 100 Gbps) (but only with 5000B MTU)

- Must balance load across many cores
  - Hard to do well

- Must aggregate packets for stack traversal
  - Linux: GSO, GRO
  - Complex mechanism (Homa impersonates TCP packets)
  - Increases receive latency
Conclusion

- Homa >> TCP
- Software protocol implementations are on their last legs
  - Impractical beyond 25 Gbps
- Need to move transport protocols to the NIC
  (needs new NIC architecture)
- Homa is ready for testing in industry; interested?