Overview of Smart NICs II: Programmable Data Planes and “Reflex Control”

Sean Choi, Muhammad Shahbaz, Balaji Prabhakar and Mendel Rosenblum
Reflex Control on SmartNICs

- Reflex Control: Applications that need *instantaneous local* response

- Motivation for using SmartNICs for Reflex Control
  - Low Latency, bump-in-the-wire hardware that exists in every server
  - Programmable Processors (NPUs) (ASIC and FPGA based)
  - Familiar Languages for Programming NPUS (P4, eBPF and C)
Netronome SmartNICs (ASIC-based)

- Programmable NPUs capable up to 100G
- Contains up to 120 Cores @ 1.2Ghz and 8GB DDR3 RAM
- Programmable via P4 and Micro-C
- Active open-source community (openNFP.org)
- Currently used to offload OpenvSwitch and other network functions
Programming Netronome SmartNICs: P4

• P4: A domain-specific language developed for expressing how the pipeline of a network device should process packets.
• P4 Abstract Forwarding Model

[Diagram showing packets moving through Packet Parser, Custom Match-Action Tables, and Packet Deparser]

• Limited language constructs for actions
• Use cases of P4 in Netronome NICs: VM Switching, Simple Firewall
Programming Netronome SmartNICs: Micro-C

• Micro-C: C-like language developed by Netronome for NPUs
• Complements P4 with Custom Actions
• Contains added constructs for accessing different types of memory
• Not a full-blown C
  • No floating point types or recursions, and limited standard library functions
• Use Cases: Deep Packet Inspection, Custom Functions
Architecture of Netronome SmartNICs

• Each NPU (Micro-Engine, or ME) is grouped into a unit called island
• Each ME has its own instructions and data, and can run 8 threads (each ME can run different programs)
• There also are shared memory and special execution units for shared data
Architecture of Netronome SmartNICs

• Netronome memories are organized into hierarchies
  • Within NPU: Registers (1 cycle), Local Memory (1 to 3 cycles)
  • Within Island:
    Cluster Local Scratch (20 to 50 cycles), Cluster Target Memory (50 to 100 cycles)
  • Outside of Island:
    Internal Memory (150 to 250 cycles), External Memory (150 to 500 cycles)
Profiling Netronome SmartNICs via Serverless Compute Workloads
Evaluation

• Type of Serverless Workloads

1. Short Workload **without** External Data Dependency
   • Simple server sending simple response

2. Short Workload **with** External Data Dependency
   • API backend making queries to remote memcached Database

3. Long Workload **with** External Data Dependency
   • Image transformer (Grayscale by averaging over RGB channels)
Architecture Overview

• Compute Architectures: (SmartNIC (λ-NIC) vs. Container vs. Bare-Metal)
Evaluation

- Experimental Testbed
  - 5x Dell R640 1U Server
  - Intel Xeon Gold 5117 14 Cores @ 2Ghz
  - 32GB DDR4 RAM
  - 120GB SSD
  - Netronome CX 10Gb SmartNIC
    - 56 Cores @ 633MHz
    - 2GB RAM
  - 10Gb Arista Switch
Evaluation

- Average Latency (ms)
  - vs. Container: 5x to 880x improvements
  - vs. Bare-Metal: 3x to 32x improvements

<table>
<thead>
<tr>
<th></th>
<th>λ-NIC</th>
<th>Bare-Metal</th>
<th>Container</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Server</td>
<td>0.05</td>
<td>1.64 (+32x)</td>
<td>45.56 (+880x)</td>
</tr>
<tr>
<td>Memcached GET</td>
<td>0.11</td>
<td>1.69</td>
<td>49.93</td>
</tr>
<tr>
<td>Image Transform</td>
<td>199.80</td>
<td>656.20 (+3x)</td>
<td>943.00 (+5x)</td>
</tr>
</tbody>
</table>
Evaluation

- Workload Throughput (up to 734x improvement)
- Measuring end-to-end time (s) for completing 10,000 Requests
  1. Sequentially by 1 thread
  2. In Parallel by 56 threads.
Conclusion

• Reflex control is needed for applications requiring fast reaction

• ASIC-based SmartNICs with enable Reflex Control

• Computes on NIC achieve huge latency and throughput gains