NICA: An Infrastructure for Inline Acceleration of Network Applications

HAGGAI ERAN†#, LIOR ZENO†, MAROUN TORK†, GABI MALKA†, MARK SILBERSTEIN†

†TECHNION – ISRAEL INSTITUTE OF TECHNOLOGY  #MELLANOX TECHNOLOGIES
FPGA-Based SmartNICs

Azure SmartNICs implementing AccelNet have been deployed on all new Azure servers since late 2015 in a fleet of >1M hosts.  [NSDI’18]
Inline Acceleration (Analogy)

https://www.pinterest.com/pin/334603447221295156/

Inline Processing - Filtering
Inline Processing - Transformation

Network ➔ SmartNIC ➔ FPGA ➔ CPU
A Key-Value Store Cache

Previous work: KV-Direct [SOSP’17], Floem [OSDI’18], LaKe [ReConFig’18]
CoAP Cryptographic Authentication
Challenges for Cloud Inline Accelerators

- No operating system abstractions
- No virtualization support:
  - performance & state isolation

The NICA infrastructure fulfills these requirements for arbitrary inline accelerators.
NICA – Contributions

• ikernel OS abstraction for inline application acceleration

• SmartNIC virtualization:
  Fine-grain time sharing & strict performance isolation
The ikernel Abstraction

AFU – Accelerator Functional Unit
Attaching an ikernel

SmartNIC

FPGA

AFU

Network

Process

ikernel

socket

AFU – Accelerator Functional Unit
Attaching an ikernel

SmartNIC

FPGA

AFU

Network

Process

Socket

ikernel

AFU – Accelerator Functional Unit
Interfaces of an ikernel

• Attach to sockets
  • Use POSIX API for data

• Custom ring – direct producer-consumer interface
  • Bypass the host network stack

• RPC – access AFU state
  • E.g. configure cryptographic keys, read counters
The ikernel Abstraction

// Create handle
k = ik_create(MEMCACHED_AFU);

// Init a socket.
s = socket(...); bind(s, ...);

// Activate the ikernel
ik_attach(k, s);

// Use POSIX APIs to receive
while (recvmsg(s, buf, ...))
  ...

107 lines of code for memcached integration
How FPGAs Are Shared [Background]

- Space sharing
- Coarse grain time sharing
- Fine grain time sharing

-Limited utilization
How FPGAs Are Shared [Background]

- Space sharing
- **Coarse-grain time sharing**
- Fine-grain time sharing

-Long context switch

AmorphOS [OSDI’18] combines the first two.
How FPGAs Are Shared [Background]

- Space sharing
- Coarse-grain time sharing
- Fine-grain time sharing

Multiple tenants share the same AFU.
+Low latency – hardware context switch (packet granularity)

NICA applies these techniques to SmartNIC AFUs
Cloud Deployment Model

<table>
<thead>
<tr>
<th>FPGA-as-a-Service</th>
<th>AFU Marketplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Customers bring their own design</td>
<td>• Cloud provider develops/audits AFUs</td>
</tr>
<tr>
<td>• Coarse-grain sharing</td>
<td>• AFUs trusted to implement fine-grain virtualization</td>
</tr>
</tbody>
</table>

- Cloud provider develops/audits AFUs
- AFUs trusted to implement fine-grain virtualization
NICA I/O Virtualization

SmartNIC

AFU

vAFU 1

NICA hardware runtime

CPU

VM

Network stack

Network
Performance Isolation

SmartNIC

- AFU
- vAFU 1
- NICA
- hardware runtime

Network

CPU

- VM
- Network stack
In the Paper

- NICA hardware runtime
- Network stack integration and TCP support
- Custom ring implementation using RDMA
- SR-IOV and para-virtual interfaces
- Implementation details
Evaluation

40 Gbps bump-in-the-wire SmartNIC:
  • Mellanox Innova Flex
    • ConnectX 4 Lx EN
    • Xilinx Kintex UltraScale FPGA

Similar to Microsoft Catapult.
Evaluation

Baseline:
- VMA user-space networking stack

Microbenchmarks:
- UDP/TCP performance
- Virtualization overheads
- I/O isolation overheads

Applications:
- memcached
- Comparison with MICA [NSDI’14]; KV-Direct [SOSP’17]
- IoT message authentication
- Node.js server
- Integrated with 20 lines of JavaScript
Memcached Cache

- Host working set: 32M keys
- SmartNIC Cache: 2M keys
- 16 byte keys and values
- Memcached UDP ASCII protocol

Simplicity of host integration:
- POSIX – 107 lines of C code
- Custom ring – 135 lines of code
Key-Value Store Cache – Bare Metal R/O

Throughput [Mtps]

Zipf distribution; Throughput by Zipf skew

At Zipf(0.99), x4 through filtering, x9 with both

Near line-rate hardware throughput

Low hit rate: x2 throughput w. custom ring
Key-Value Store Cache - Virtualization

- 1 core, 5 GB host RAM, 2M key cache, Zipf(0.9)
Latency under virtualization

- 60% hit-rate – 2.1 µs
  - same as bare-metal
- Misses on the CPU – 12-62 µs
  - Compared to 6 µs on bare-metal
- Negligible sharing overhead
Key-Value Store – Performance Isolation

Throughput over time

Throughput [Mtps]

VM 1
VM 2
VM 3
Conclusion

• NICA framework enables SmartNIC inline processing in cloud environments.
• Find our code at https://github.com/acsl-technion/nica

Thank you!
Questions?