Consolidating Network Resources with Rack-Level Network Pooling

Will Lin† Ryan Kosta† Arvind Krishnamurthy* Yiying Zhang
w5lin@ucsd.edu rkosta@ucsd.edu arvind@cs.washington.edu yiying@ucsd.edu
UC San Diego * University of Washington †Student

Resources in today's data-center network devices are largely under-utilized because of two factors. First, many SmartNICs and programmable switches are bloated with features that are not utilized in the common case [*Calfield-HPSR'18, Firestone-NSDI'18, Bansal-NSDI'23, Wang-HotCloud'20*]. Over time, vendors keep adding more features to their network device products to meet the diverse requirements of different customers, resulting in significant resource waste. Even a complex application consumes only a small fraction of resources (*e.g.*, NetChain [*Jin-NSDI'18*] uses 3% MAT resources). Second, network traffic load varies significantly over time in data centers. We analyze two sets of traces: a Facebook trace that consists of Web, Cache, and Hadoop workloads [*Roy-SIGCOMM'15*], and an Alibaba trace that hosts latency-critical and batch jobs together [*Guo-IWQoS'19*]. Both data centers exhibit high burstiness, and bursts happen at different times for different end hosts.

Furthermore, we observe that network traffic peaks at different times across end hosts in the data center, which means that the peak of aggregated traffic (*i.e.*, *peak of sum*) is lower than the sum of peaks at individual end hosts. We analyzed the Facebook and Alibaba traces to calculate the peak of traffic aggregated at the rack level and at the data-center level. They are orders of magnitude lower than the sum of individual peaks, as shown in Figure 1.

Based on these observations, we believe network hardware resources should be *consolidated*. We propose to consolidate network devices that used to be attached to individual end hosts in a rack into a separate *network pool*. Specifically, each network device in the pool (called *sNIC*) connects to a small set of end hosts and to the ToR switch. Additionally, we connect all the sNICs in the pool with an internal topology such as a ring or torus, as shown in Figure 2. By connecting sNICs as a pool, we can offload processing needs from one sNIC to another sNIC when the former is overloaded, thereby achieving *rack-level network consolidation*.

We provide two types of consolidation: traffic consolidation and network processing consolidation. The first consolidates traffic from end hosts at the sNIC they connect to. Because different end hosts' traffic streams peak at different times (from our analysis), the consolidated traffic that is then sent to the ToR switch has a much smaller peak than the sum of peaks. Thus, the link bandwidth between an sNIC and the ToR switch can be much lower than the sum of bandwidths for the links between the sNIC and the end hosts it connects to. The second type of consolidation is about network processing. As different workloads can require different types of network functions at different times, we can consolidate the network

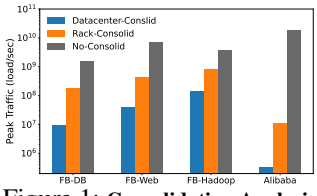


Figure 1: **Consolidation Analysis of Facebook and Alibaba Traces.**

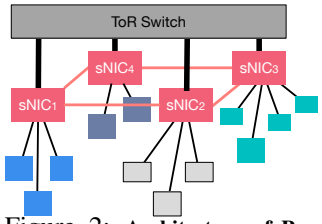


Figure 2: **Architecture of Proposed Network Pool.**

functions needed by all the end hosts to sNIC. Furthermore, when built with FPGA or software, we can perform time sharing to use the same sNIC hardware resource for different network functions at different times. Overall, these types of consolidation allow for a *lower cost to build, run, and manage network resources in a rack*.

To realize efficient consolidation, we need to solve several challenges. The first challenge is in balancing load across sNICs and routing traffic afterwards. We prioritize the local sNIC that is directly connected to the end hosts. If it is overloaded, we use a Weighted-Load-Balancing-based [*Singh-SPAA'02*] algorithm that favors shorter routes and sNICs that are less loaded but are already running the needed network functions. Afterwards, we install forwarding rules in sNICs along the route. We monitor load at each sNIC using a set of leaky bucket counters to capture user-intended flow throughput (not the actual throughput achieved). We use a global controller to collect the monitored load and run the route-selection algorithm.

The second challenge is in efficient sharing of sNICs' hardware resources for dynamic load and network functions, while guaranteeing the performance of each function. We confront this challenge by building sNIC with FPGA, with a set of optimizations for fast network function spawning, switching, and migration.

The third challenge is in the handling of sNIC failures. Since multiple end hosts rely on a single sNIC, failures can affect more devices. To mitigate this, we connect each end host to two sNICs. This architecture also enables better load balancing under load spikes.

We prototype the sNIC device on two HTG9200 FPGA boards, each with nine 100 Gbps ports. We also evaluate larger scale with simulation built based on ns-3. A recent work, Sirius [*Bansal-NSDI'23*], also adopts the general idea of network resource consolidation but uses a different approach. They build specialized racks to handle network processing tasks like firewalls for regular racks. As such, all traffic needs to go through the additional hops to the specialized racks. Besides, they also cannot achieve traffic consolidation and lack support for rich sets of network functions.

Consolidating Network Resources with Rack-Level Network Pooling

UC San Diego WukLab

Will Lin†, Ryan Kosta†, (†Student)
Arvind Krishnamurthy, Yiying Zhang

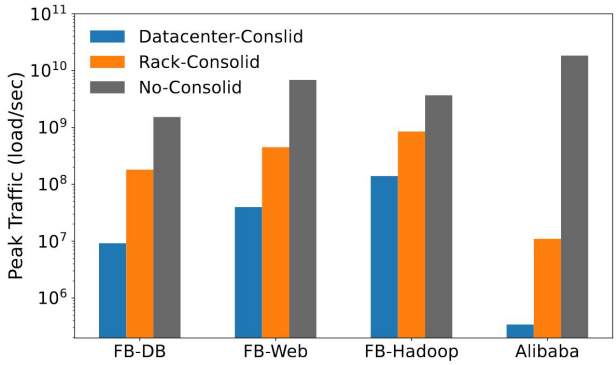
PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

Motivation

Network devices are under-utilized.

- Many bloated with underutilized features
- Network traffic load fluctuates significantly *[Roy-SIGCOMM'15, Guo-IWQoS'19]*

=> Can largely benefit from consolidation

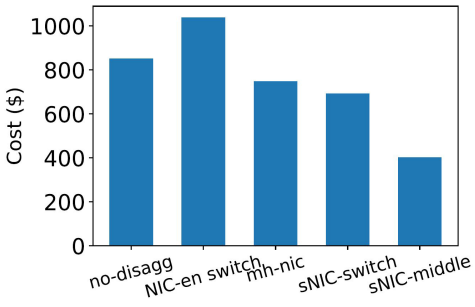


Facebook: Web, cache, and Hadoop trace

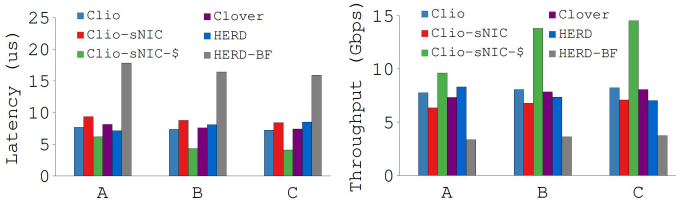
Alibaba: Co-hosted real-time and batch jobs trace *[Roy-SIGCOMM'15, Guo-IWQoS'19]*

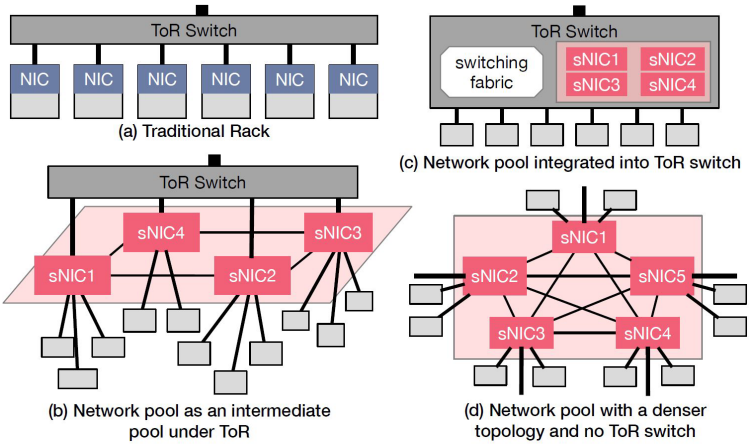
Initial Results

- CapEx cost of different architectures



- Hardware prototype performance
  - End-to-end eval on *[Clio-ASPLOS'22]*





Proposal

Consolidate & pool network functions

- A pool of network devices (called sNIC)
  - Sit in the middle of a rack for end hosts
  - Connected with an internal topology
- Network function offloaded to the pool
  - Functions and traffic consolidated
  - Load balanced across sNICs

Implementation and Future Works

Implementation
- Prototype sNIC on HTG-9200 FPGAs
- Simulate rack topologies in NS-3

Efficient and fair sharing
- Multi-tenancy support for hetero resources

Fault tolerance
- Avoid single point of failure
- Managing stateful network functions

Load balancing
- Balance load while controlling latency