An Infrastructure for the Development of Kernel Network Services.

Proof of Concept: Fast UDP

Motivation
Performance degradation of HPC applications is caused by several factors:

- Host processor overhead due to communication processing
- Memory latency on inbound network data
- Cost of splitting OS functionality between host and NIC
- Data placement overhead (memory copies)
- Overhead due to external interrupts

Poor interaction of the NIC with the OS and applications, leading to poor performance

Goal
Build an infrastructure to:

- Study NIC/OS/Application interaction
- Cache Injection
- OS and Hypervisor bypass
- Protocol Offloading
- Interrupt direction and filtering
- Develop and evaluate next-generation NICs

Network Infrastructure
Framework to create simulated NICs

- Run arbitrary functionality
- Created as dynamic libraries
- Plug-in to IBM’s Mambo full-system simulator
- Interact with host through the Shim Layer:
  - Provides the glue between NIC and host
  - Simulated NIC is developed without the need of Mambo source code
- Entry points explicitly defined by the Shim Interface

The Shim Layer

Shim Interface
- mem_write, cache_write
- mem_read, cache_read
- memmap_define
- memmap_delete
- set_memmap_io_funcs
- schedule_job
- delay_cycles
- raise_interrupt

Cache Injection

Fast UDP
- Splinter data from control information
- Application's data bypasses the OS
- Delivery notification provided by the OS
- Matching on the NIC
- NIC has enough information to perform data placement directly
- NIC Offload
  - Splintering, Message Matching, UDP/IP checksum semantics

User / Kernel / NIC Interfaces

NIC Data Structures and Matching

Results

Conclusions and Future Work
- Developed an infrastructure to:
  - Better understand the interactions between smart NICs, the OS, and applications
  - Study recent and future NIC architectures
  - Make a case for kernel network services that improve application's performance
- Proof of concept: Fast UDP
  - 5% improvement on an 80% computation-bound application
- Future Work
  - Study OS services to leverage cache injection for HPC applications
  - Study functionality placement of these services between NIC and OS

Test Application

```c
while(1) {
    recvfrom(sock, buff+offset, ...);
    if (offset >= i++ * chunk_size)
        sort_chunk(buff, i);
    if (i == num_chunks)
        break;
}
```