Scalable and adaptive network stack architecture

Pravin Shinde

Systems Group

ETH

PhD research proposal
(work in progress)

The Problem

Traditional monolithic network architectures do not utilize the Network Interface Card(NIC) hardware fully in current setup!

The Past

The beginning

- Single core machines
- Simple network card with basic RX/TX features
- Result: Monolithic network stack

The Present

Current setup

- Multicore machines
- Advanced Network cards
  - Checksum offloading
  - TCP offloading
  - Receive side scaling
  - Remote Direct Memory Access (RDMA)
- Result: Network stack with data parallelism
  - Packet level parallelism
  - Connection level parallelism
- Ignore the additional hardware features: The Linux way!

Hardware features are complicated

- Vendor specific set of features
- Vendor specific interfaces
- No standardization yet
- Is it worth, yet?

Yes! Hardware features can be useful!!

- Improves performance
  - Routebricks
  - Safecard gigabit IPS
  - Packetshedder
- Added complexity in traditional monolithic stack

But, the trend is towards more hardware features

The Future

The trend

- More advanced and virtualizable NIC's
  - Multiple RX/TX queues
  - Hardware filters
  - Interrupt routing
  - Direct cache access
  - Programmable cores
- Manycore machines
- Result: ????

The Hypothesis

If network stack is organized as fine grained **Protocol Graph**, then available hardware resources can be better used by efficiently *mapping them on appropriate hardware resources.*

Protocol graph based network stack architecture

History

- X-kernel: To support new protocols easily
- Dynamic Protocol graphs: Adaptable to application requirements
- Scout: Protocol graphs to exploit global knowledge
- Click: Customizable and extensible
- Streamline: Protocol graphs to avoid unnecessary data copy

Why Protocol graphs?

- Ability to decompose the stack into small units
- Ability to map these units on available hardware features
- More units provide better control on scheduling
- More units give deeper pipeline
- Transformations and optimizations from query processing or graph theory can be applied
- Ability to transparently emulate the units in software when they are missing in hardware

Drawbacks of protocol graphs

- Increased communication
- Increased overhead on scheduling

Proposed design

Fine grained protocol graphs for adaptability and scalability

Proposed design: Step-1

Logical Protocol Graph

- Capture the application requirements
- Create a logical protocol graph which can optimally meet the application requirements

Research questions

- What should be the interface between an application and network stack?
- What should be the granularity of decomposing network stack?

Proposed design: Step-2

System Knowledge Base(SKB) maintains the information about

- Properties of system resources
- Current allocation and utilization of system resources

Proposed design: Step-3

- Logical protocol graph to physical resource allocation plan

Research questions

- What optimizations/transformations can be applied on logical protocol graphs?
- Can mapping algorithm scale with increasing number of applications?
- Incremental optimal resource allocation for changing requirements?

How to evaluate the **adaptability** of a network stack architecture?

Possible evaluation approach: Adaptability

Comparing the optimized solution with adaptable solution

- Compare *performance difference* against *code complexity*
- How easily and efficiently one can utilize
  - Programmable cores on NIC
  - GPU's
  - Accelerators
  - FPGA

Possible evaluation approach: Scalability

Scalability

- With number of Cores
- of mapping algorithm with applications
- With number of NIC's

Possible evaluation approach: Application specific

Example: 5 Micro-second RPC

- Without hand-tuning
- Without static resource allocation
- Automatically by the mapping of protocol graph

Conclusion

- Monolithic network stacks are too rigid
- Fine grained protocol graph design can provide more flexibility
- It is not clear yet if it is just too complicated to use all these hardware features?