CS 5413 Group Projects

Friday, February 17, 2017

Goals

- Identify a fun and challenging semester project
- Suggested project ideas relate to:
  - Disaggregated Datacenters
  - Designing new Dataplane Programming applications
  - Containers in the Cloud
  - Model Cloud Performance
- Or, can suggest your own project

- Submit and Present at BOOM, Mar 29 and Apr 19
- End of semester presentation

Timeline

- Feb 17: Identify a Project
- Feb 27: Propose a Project
  - Identify what you will do, who will do what, and dates

- Mar 22: Intermediate Project Report
- Mar 29: BOOM Submission Deadline

- Apr 12: Intermediate Project Report II
- Apr 19: BOOM Presentation

- May 10: Final Presentation/Demo Day

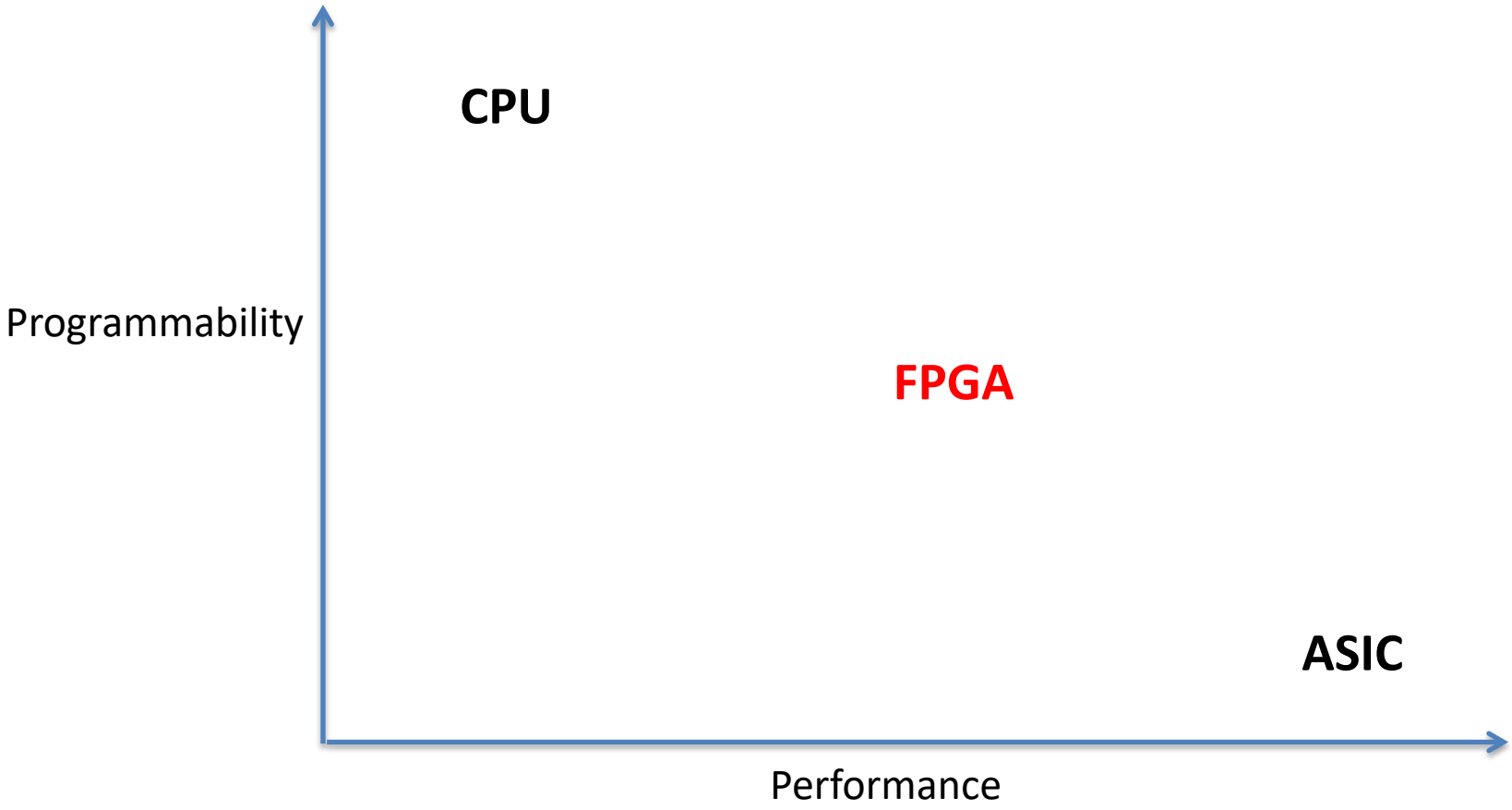
Implementing RDMA-like Protocol on FPGAs to support Memory Disaggregation

Vishal Shrivastav, Cornell University

Why Memory Disaggregation?

- More efficient and fine-grained resource provisioning
- Circumventing “Memory Wall”/”Power Wall”
- Allow each resource technology to evolve independently

Why FPGAs?



What will you do?

- Implement a DMA engine on FPGA
- Design a simple protocol for remote memory communication
  - read_mem(addr)
  - data = read_response()
  - write_mem(addr, data)

Skills Required

- Systems and Networking knowledge
  - DMA, Network stack etc.
- Familiarity with FPGA development tool chains
  - Preferably **Altera** tool chain
- Knowledge of hardware programming language
  - Preferably **Bluespec System Verilog (BSV)**

Outcomes*

- Get experience building a real system on FPGAs
  - a very sought-after skill in networking industry at the moment
- Submit your work to BOOM and maybe even win
- Get an A+ in the course
- Get a research publication in the near future

**subject to good performance*

Thank you !

P4 and P4FPGA

Dhruv Singal

Implement Network Hardware-based Paxos

Problem: Traditional Paxos is usually implemented in software and thus is both slow and unpredictable in terms of time consumed. Implementing it, at least partially, in hardware can lead to improved performance and predictability.

Task: Implement Paxos using the P4FPGA framework in the network stack

Outcomes: Produce an implementation of Paxos that runs on NetFPGAs

References:

- Wang, H., Soule, R., Dang, et al (2017). *P4FPGA : A Rapid Prototyping Framework for P4.*
- Dang, H. T., Canini, M., Pedone, F., & Soulé, R. (2016). *Paxos Made Switch-y.*
- Dang, H. T., Sciascia, D., Canini, M., Pedone, F., & Soulé, R. (2015). *NetPaxos: Consensus at Network Speed.*

Implement Time Synchronization in P4PFGA

Problem: Current clock synchronization protocols in datacenter networks such as NTP and PTP are affected by the characteristics of packet switching networks such as network jitter, packet buffering and scheduling in switches, etc, which must be accurately measured to synchronize clocks precisely. DTP solves this.

Task: Implement the Datacenter Time Protocol using the P4FPGA framework

Outcomes: Produce an implementation of DTP that runs on NetFPGAs

References:

- Wang, H., Soule, R., Dang, et al (2017). *P4FPGA : A Rapid Prototyping Framework for P4.*
- Lee, K. S., Wang, H., Shrivastav, V., & Weatherspoon, H. *Globally Synchronized Time via Datacenter Networks.*

Thank you !

Introduction to X-Containers

Zhiming Shen

Docker

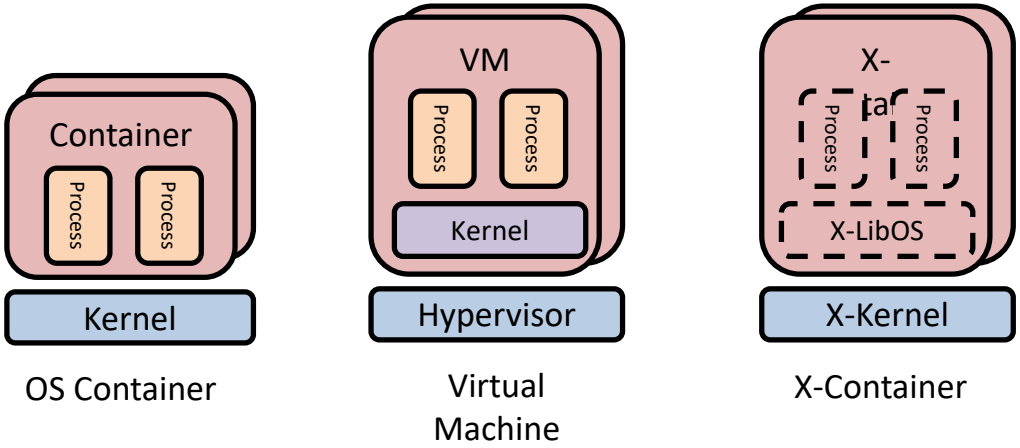
- Portability
  - Packaged once, run everywhere
- Efficiency
  - Light-weight OS-level virtualization

The Problems

- Weak isolation
- Kernel compatibility
- Kernel customization

X-Container

- Run each container with a dedicated kernel
- Break isolation between the kernel and applications



Implementation

- X-Kernel:
  - Based on Xen, a type-1 virtual machine monitor
- X-LibOS:
  - Based on para-virtualized Linux kernel
- Connected to Docker images automatically
- Requirements:
  - C programming
  - Linux Kernel, virtualization, Xen hypervisor
  - Docker, Linux file systems etc.

Integrating X-Containers with Docker Engine

- Goal:
  - Create a new Docker engine that uses X-Containers as backend
- Tasks:
  - Understand the design of Docker engine
  - Implement a new backend for Docker engine that uses X-Containers
  - Support common features such as shared file system and pipe redirection
- Reference:
  - Docker engine source code: https://github.com/docker/docker

DPDK-Optimized X-Containers

- Goal:
  - Optimize X-Container architecture with DPDK
- Tasks:
  - Integrate DPDK-based open-vswitch with X-Containers
  - Run DPDK applications inside X-Containers
  - Optimize DPDK performance
- Reference:
  - DPDK: http://dpdk.org/
  - Use Open vSwitch with DPDK: https://software.intel.com/en-us/articles/using-open-vswitch-with-dpdk-for-inter-vm-nfv-applications

X-Container with PVH mode

- Goal:
  - Optimize memory virtualization with hardware assisted paging
- Tasks:
  - Port X-Container architecture to the latest version of Xen
  - Enable X-LibOS in PVH mode
  - Measure the performance of PVH-based X-Containers
- Reference:
  - Xen PVH mode: https://wiki.xen.org/wiki/Xen_Project_Software_Overview#PVH

Shared File System between the Host and X-Containers

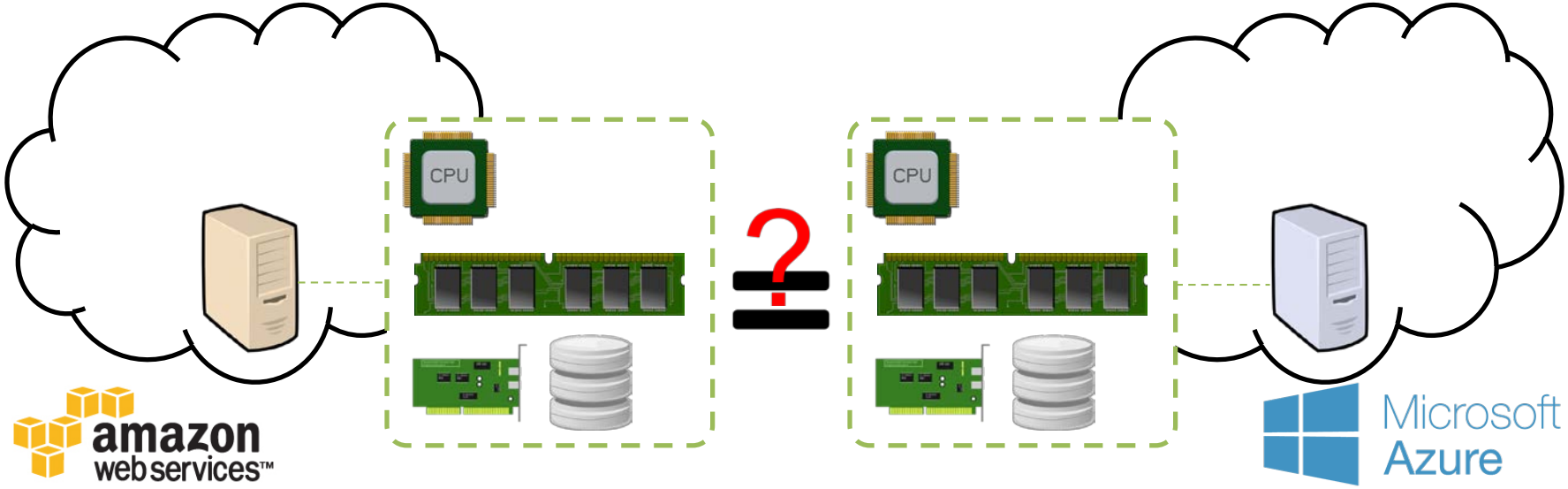
- Goal:
  - Efficient file sharing between X-Containers and the host
- Tasks:
  - Understand Xen grant table and memory sharing
  - Implement a new file system based on Xen grant table
  - Performance test
- Reference:
  - Xen grant table: https://wiki.xen.org/wiki/Grant_Table
  - Linux file system implementation: http://pages.cs.wisc.edu/~remzi/OSTEP/file-implementation.pdf

Thank you !

The Cloudmodel Project

Weijia Song

The Problem



No, they are not equal

- Different Hardware
- Different resource sharing level
- Different Hypervisors
- and More…
- How should a Cloud user choose the Cloud service for its application?

Cloudmodel is a tool measuring the resources in the Cloud

- A set of micro benchmarks to quantify the resources. (**C/C++**, **ASM**)
- **Python** scripts automatizing the measurement/visualization.
- Github Repo: https://github.com/songweijia/cloudmodeling
- Current Progress: CPU cache size/throughput/latency, memory throughput/latency.

Tasks

- Task 1: Design and implement the disk I/O and file system performance modeling/evaluation module.
- Task 2: Design and implement the network modeling/evaluation module.
- Task 3: Build a ***cloudmodel*** service to automatize the modeling/evaluation/visualization.

Thank you !