Disaggregated Operating System

Yizhou Shan, WanEih Huang
shan13, huan1031@purdue.edu

Disaggregated Datacenter

For many years, the unit of operation and failure entity in datacenters has been a monolithic computer, one that contains all the hardware resources (typically a processor, some main memory, and storage devices) that are needed to run user programs and runs an OS (or hypervisor) to manage these hardware resources.

Recently, there is an emerging trend to move towards a disaggregated hardware architecture that breaks monolithic servers into independent hardware components that are connected to a fast, scalable network components. Each component will have its own controller to manage its hardware and can communicate with other components through a fast network.

Disaggregated Operating System

OSes built for monolithic computers can not handle the distributed nature of disaggregated hardware components. Datacenter distributed systems are built for managing clusters of monolithic computers, not individual hardware components. When traditional OS operations spread across hardware components over the network, these distributed systems fall short. Clearly, we need a new operating system for the disaggregated datacenter architecture.

We propose the concept of disaggregated operating system for the disaggregated datacenter architecture. The basic idea is simple: *When hardware is disaggregated, the operating system should be also*. A disaggregated OS breaks operating system services into micro-OS services, or component managers, and runs them on different hardware components. These component managers can be heterogeneous and can be added,

restarted, or reconfigured dynamically without affecting the rest of the disaggregated system.

Challenges

There are two main challenges in building a disaggregated OS: How to cleanly separate OS services and map them to hardware components? Different hardware components have different constraints. And how to ensure failure independence? Since an application running on disaggregated architecture can be using a set of hardware components and one component can host multiple applications, a component failure should not affect the running state of applications.

Lego

We are building Lego, a distributed, loosely-coupled, failure-independent OS, designed and built from scratch for disaggregated hardware architecture. Lego runs a component manager at a hardware component and appears to applications as a normal distributed system. It consists of three types of component managers: processor manager, memory manager, and storage managers.

To cleanly separate components, we will build each manager as a *stateless service* and different managers communicate with requests that contain all the information needed to fulfill them.

To evaluate Lego, we plan to emulate hardware components using commodity monolithic servers. For example, to emulate a memory component, we will only enable one or two cores of a server, while to emulate a processor, we will limit the accessible physical memory of a server.

Current Status

We have finished low-level booting code, memory management, process management, and IB network stack. Currently we are building specific functionalities for processor, memory components. In the early stage of Lego, storage components will be built in user-level to serve requests.