

**Stony Brook University**  
**The Graduate School**  
Doctoral Defense Announcement

**Abstract**

End-to-End Abstractions for Application-Aware Storage

By

**Gopalan Sivathanu**

Modern computer systems are a composition of several logically independent layers. From a very simple hardware-software layering in early days, this layering in computer systems has increased significantly over time in both its depth and complexity. For example, in the past, file systems communicated directly with the disk hardware, whereas today layers such as logical volume managers, RAID, or even a network can exist in between. Although providing many important benefits, this rampant layering has also led to the well-explored problem of information-divide in the systems stack. Layers hide information, thus constraining functionality and limiting the power of individual layers. A particularly striking instance of this general problem exists in the storage stack today. Modern high-end storage systems have significant processing capabilities, but despite their potential, storage systems are constrained in their functionality because they are oblivious of knowledge about higher layers such as the applications using them.

In this thesis proposal, we seek to answer a simple question: *how can we convey application-level information across the diverse modern storage stack in a simple and generic manner?* We propose two flexible abstractions to solve this problem. The first abstraction we present is the notion of *type-awareness* in the storage stack. In type-aware storage, lower layers of the storage stack such as the disk are aware of the pointer relationships between disk blocks that are imposed by higher layers such as the file system. Type-awareness enables semantics-aware optimizations in the lower layers of the storage stack, and also active enforcement of invariants on data access based on the pointer relationships, resulting in better security and integrity. The second abstraction we evolve is Context-Aware I/O (CAIO), a generic mechanism to propagate application context end-to-end through the storage stack. CAIO provides generic interface to convey *application-data* and *application-I/O* relationships to the storage stack, enabling interesting functionality.

Through several case studies, we demonstrate the flexibility and benefits of both abstractions and show that they present a simple yet effective general interface to build the next generation of storage systems.

**Date:** February 22, 2008

**Time:** 11 AM

**Place:** CS 2311

**Program:** Computer Science

**Dissertation Advisor:** Dr. Erez Zadok