

# Securing Applications using Operating System Transactions

and

File System Security Research Overview

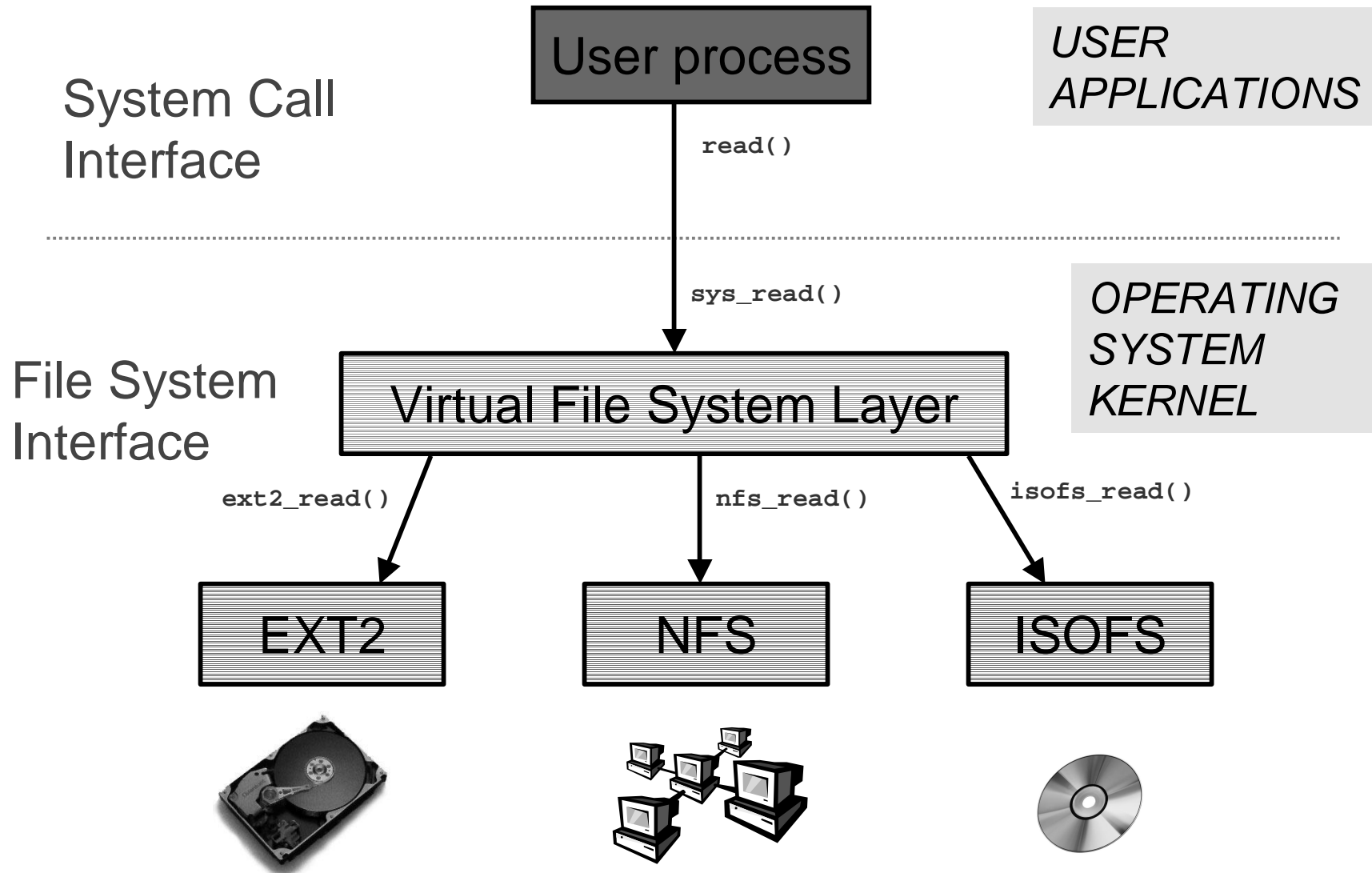
**Erez Zadok**

File-systems and Storage Laboratory

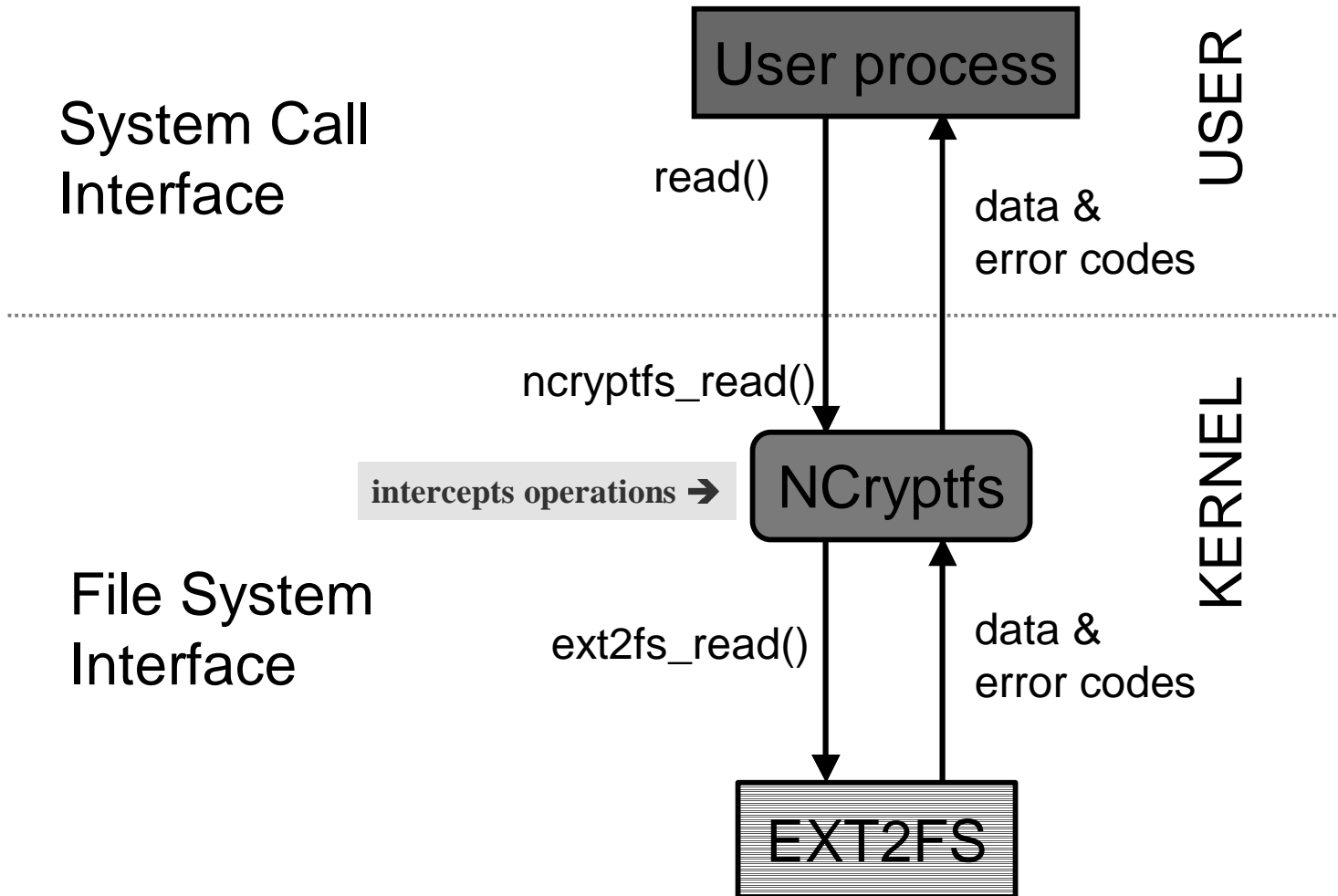
Stony Brook University

<http://www.fsl.cs.sunysb.edu/>

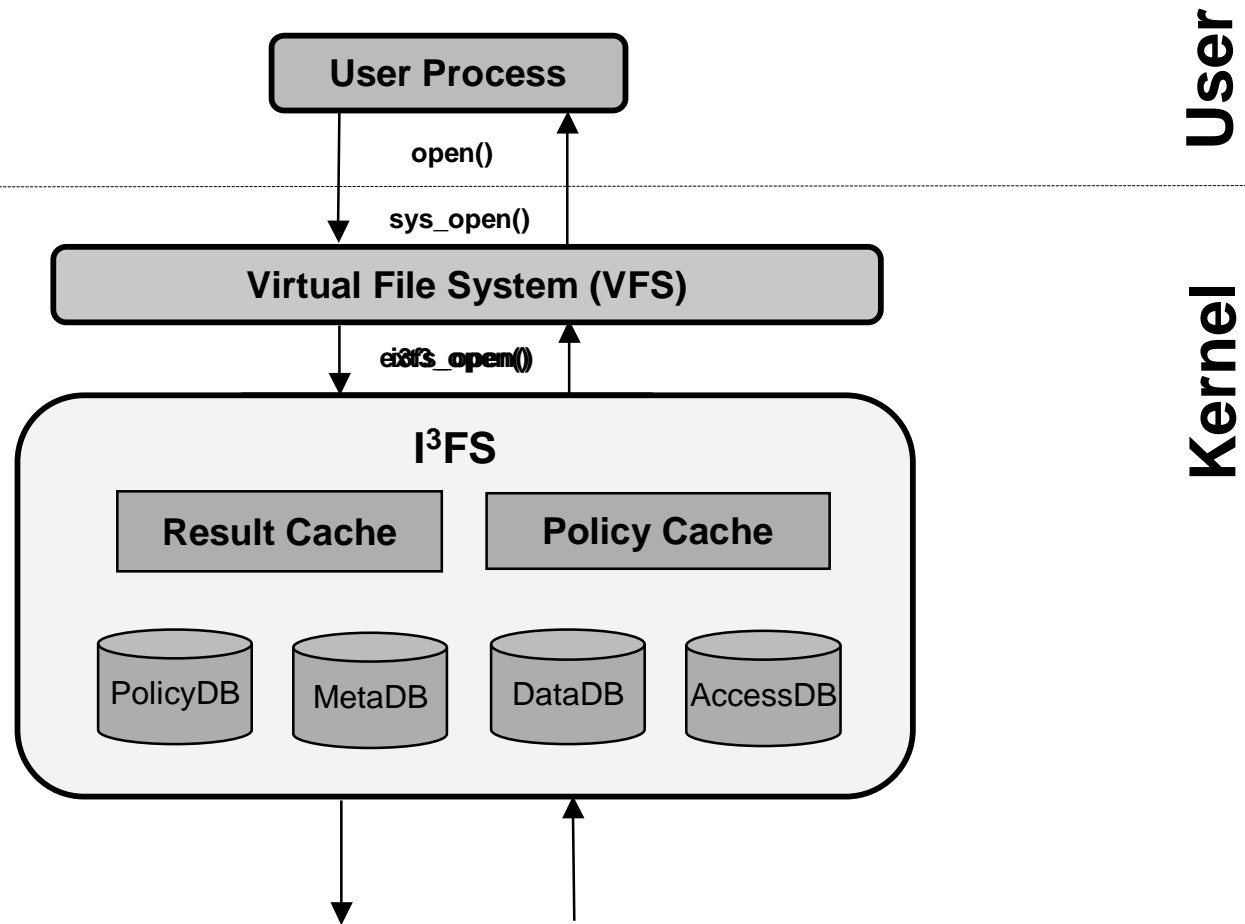
# How File Systems Work



# File System Interception



# Integrity-Checking



# Avfs: Anti-Virus File System

System Call Interface

User process

read()

data & error codes

avfs\_read()

scanbuf()

Avfs

Oyster

File System Interface

ext2\_read()

data & error codes

EXT2

Virus DB

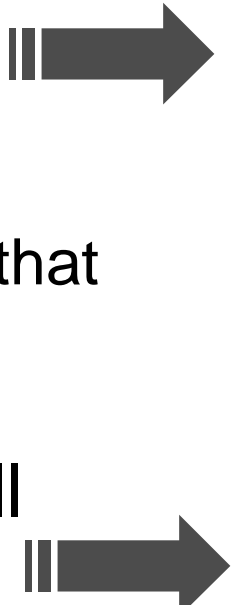
USER

KERNEL

# Secure File System Projects

File System	Benefit
NCryptfs**	Encryption: protect data privacy [Usenix, SISW]
I3FS	Detect unauthorized data changes [LISA]
Antivirusfs	Transparent anti-virus checking [Usenix Security]
Tracefs	Fine-grained monitoring (IDSs) [FAST]
Replayfs	Fine-grained replaying (forensics) [FAST]
Versionfs, Snapshotfs	Checkpoint file state, easy undo and redo (forensics) [FAST]
RAIF	Distributed Storage Survivability [ClusterSec**]
SDFS	Secure Deletion of files [SISW]
E2EC**	End-to-End security using NFSv4 (IBM)

# POSIX API Problems

- Guarantees only single system call behavior
    - ◆ but not multiple system calls
    - ◆ OS may not guarantee even that
  - Other system call can interleave
  - System failure leave application state inconsistent on disk
    - ◆ fsck won't help you
  - TOCTTOU security exploits
- 

# Solution? Databases

*Applications can use databases:*

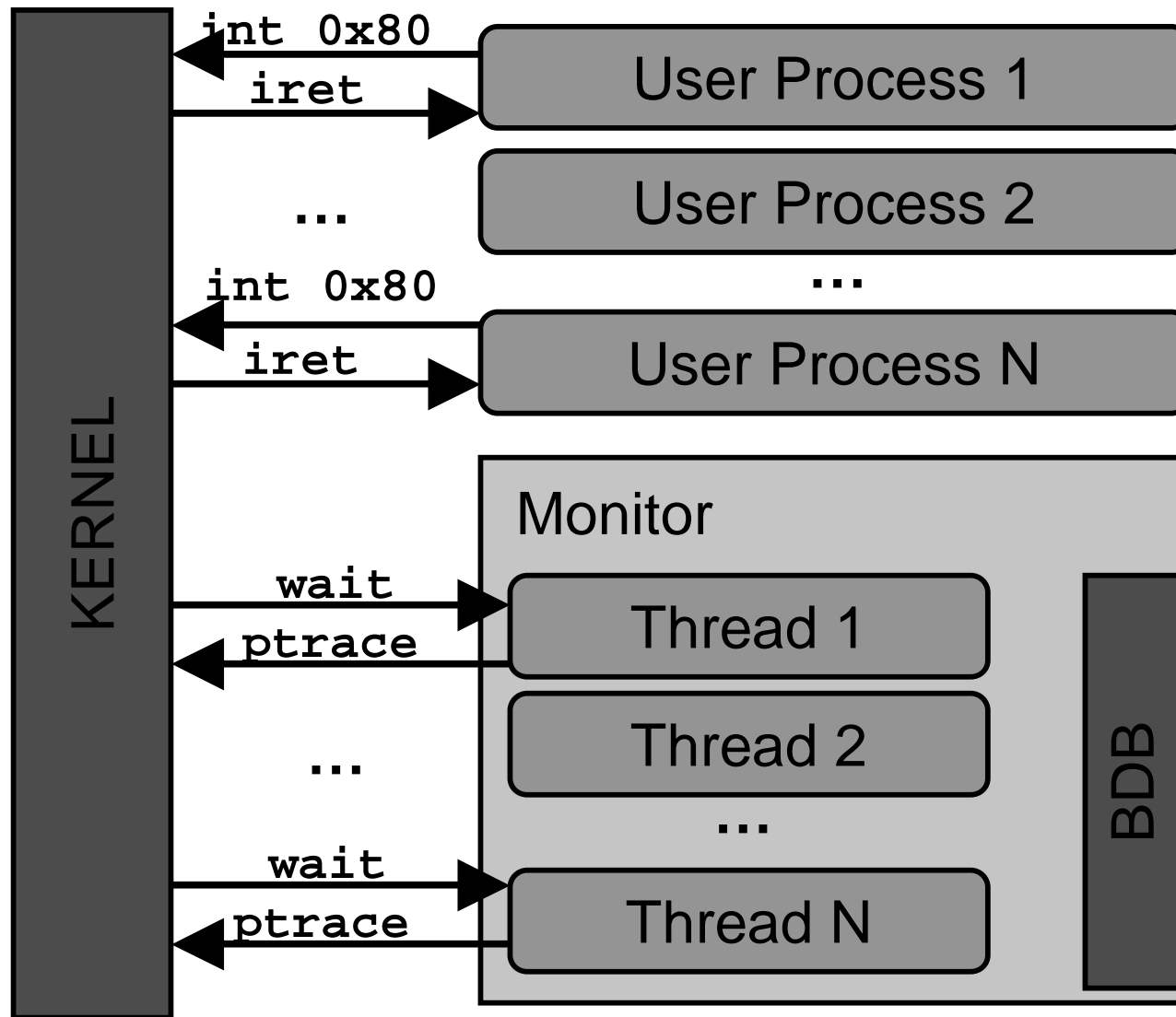
- ☺ Offer full ACID semantics
- ☹ Cumbersome to use
- ☹ Differing APIs
- ☹ Bloated: SQL, stored procedures, query optimizations, etc.
- ☹ Each application has to be modified

# Our Solution: Berkeley DB

*Embed BDB into operating system:*

- ☺ Offer full ACID semantics
- ☺ Smaller code base (150k LoC)
- ☺ Maintains POSIX API
- ☺ No need to change applications
- ☺ Easy to use, lightweight
- ☹ Kernel coding is hard

# Current User-Level Prototype



# Example Advantages

- Sample applications we modified
  - ◆ `mail.local` (part of sendmail)
    - reduced code size 450 to 78 (~6x smaller)
  - ◆ `cvS`
    - 100+ LoC of “wannabe” code
  - ◆ `tar`
    - doesn't even try
- Easy to add begin/end transaction calls
  - ◆ Code more reliable
  - ◆ Code more secure
  - ◆ Code easier to audit
- *Profiles* for legacy application

# Proposal: Move to Kernel

