

# WatDFS

## A Project for Understanding Distributed Systems

**Michael Abebe**

Brad Glasbergen

Khuzaima Daudjee

[tiny.cc/watdfs](https://tiny.cc/watdfs)

SIGCSE 2019



UNIVERSITY OF  
**WATERLOO**

File Explorer window showing the Quick access view. The ribbon includes File, Home, Share, and View tabs. The ribbon buttons are grouped into Clipboard, Organize, New, Open, and Select.

The address bar shows the current location: Quick access. A search bar for Quick access is also present.

The left sidebar shows the navigation pane with the following items:

- Quick access
  - Desktop
  - Downloads
  - Desktop
  - Desktop
  - Screen
  - Windows Technic
- OneDrive
- This PC
- Local Disk (C:)
- Network
- Homegroup

The main area displays:

- Frequent folders (6)**
  - Desktop This PC
  - Downloads This PC
  - Desktop Shared Folders (\\vmware-host)...)
  - Screen Local Disk (C:)\Windows\Web
  - Desktop \\vmware-host\Shared Folders
  - Windows Technical Preview Local Disk (C:)\Windo...\Wallpaper
- Recent files (15)**
  - ssh1.png Shared Folders (\\vmware-...\Desktop)
  - windows-wallpaper-lock-screen.png \\vmware-host\Shared Fol...\Desktop
  - windows-wallpaper-control-panel.png \\vmware-host\Shared Fold...\Desktop
  - windows-wallpaper-location2.png \\vmware-host\Shared Fol...\Desktop
  - windows-wallpaper-location.png \\vmware-host\Shared Fol...\Desktop
  - ssh1.png \\vmware-host\Shared Fol...\Desktop
  - img100.jpg Local Disk (C:)\Windows\Web\Screen
  - ssh1.jpg \\vmware-host\Shared Fol...\Desktop
  - ssh1.jpg \\vmware-host\Shared Fol...\Desktop
  - img1.jpg Local ...\Windows Technical Preview
  - ssh4.tif Shared Folders (\\vmware-...\Desktop)
  - ssh3.tif Shared Folders (\\vmware-...\Desktop)

21 items



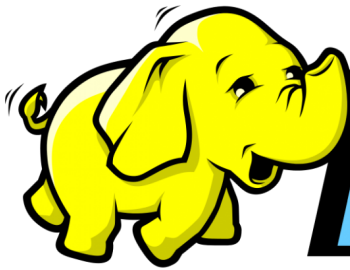








*Sun*<sup>®</sup>  
microsystems



**hadoop**



**ceph**



# Build Your Own Distributed File System



# Project Goals

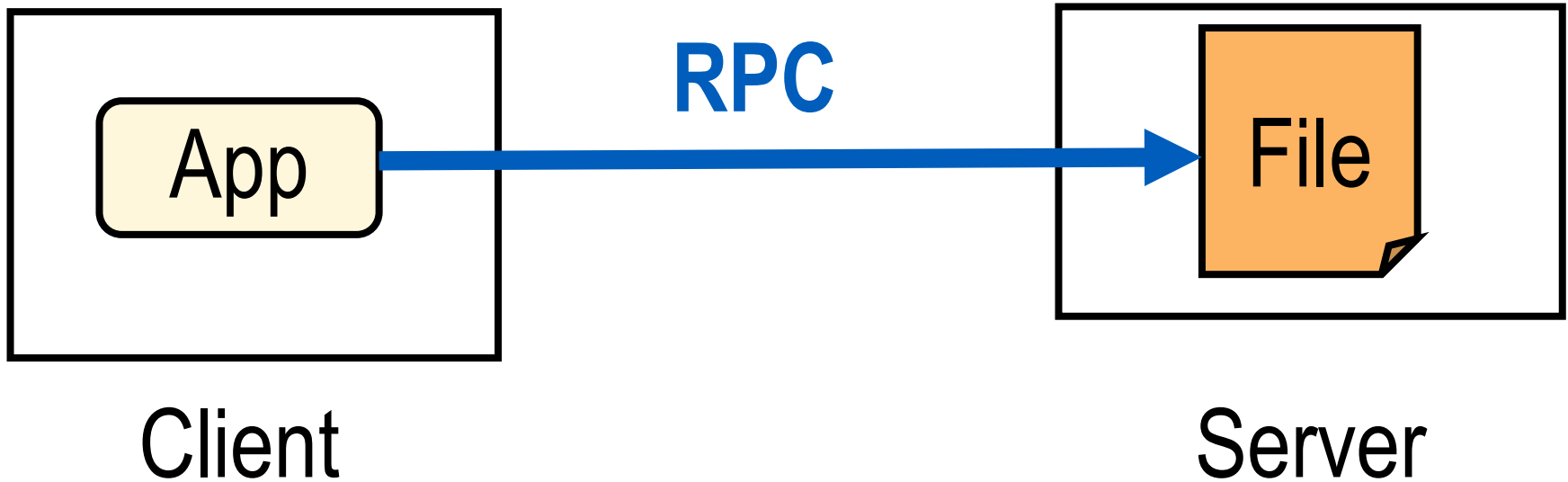
Cover wide range of course material

Interaction with **common** systems & applications

Provide **high-quality** and **timely feedback**



# Distributed File Systems

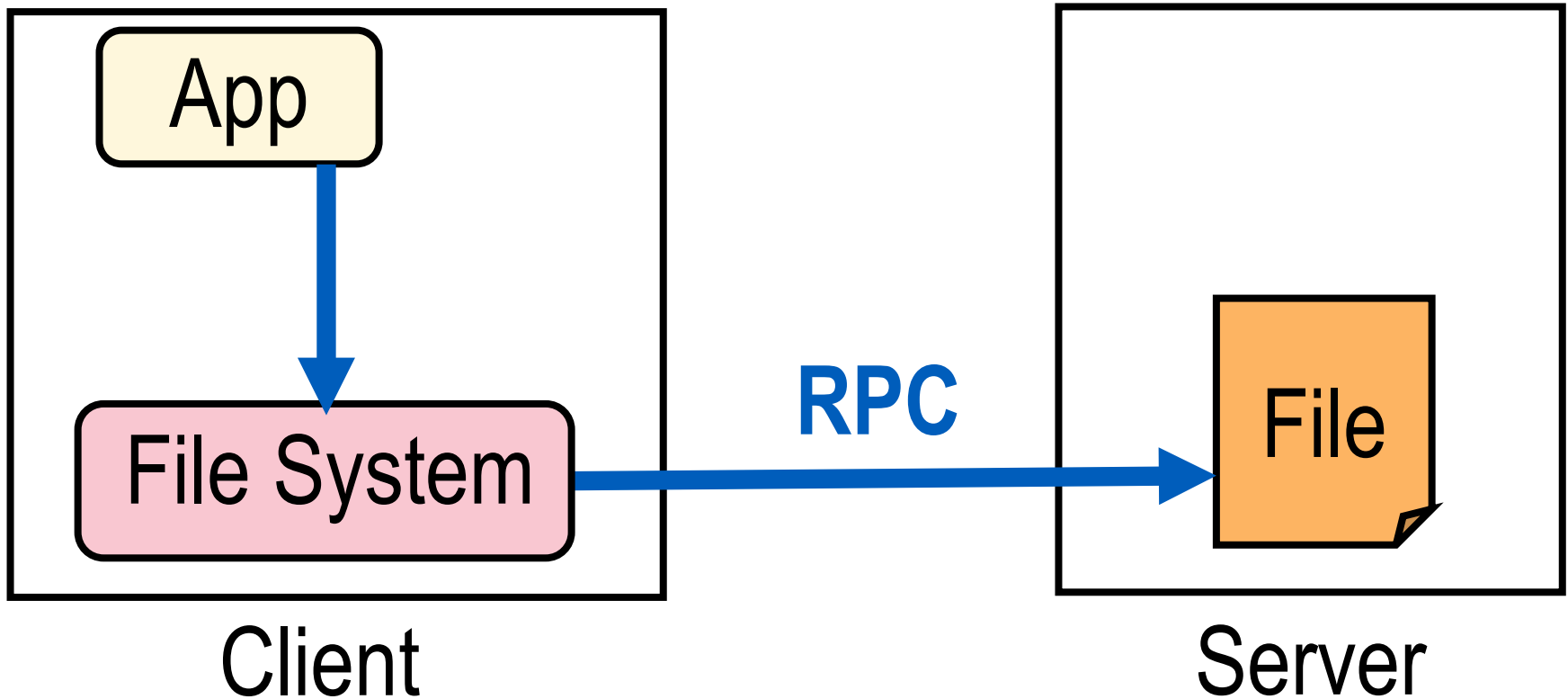


# Distributed File Systems



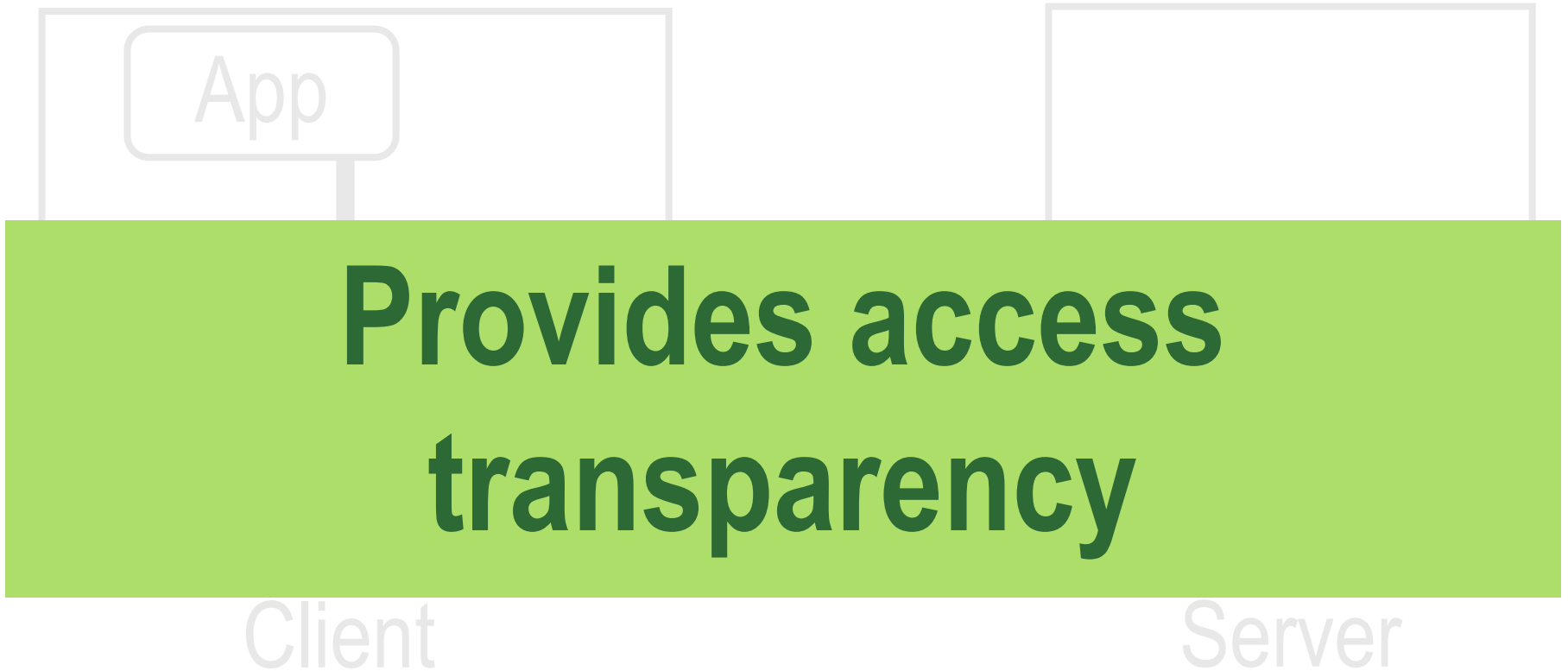
**Lacks access transparency**

# Distributed File Systems

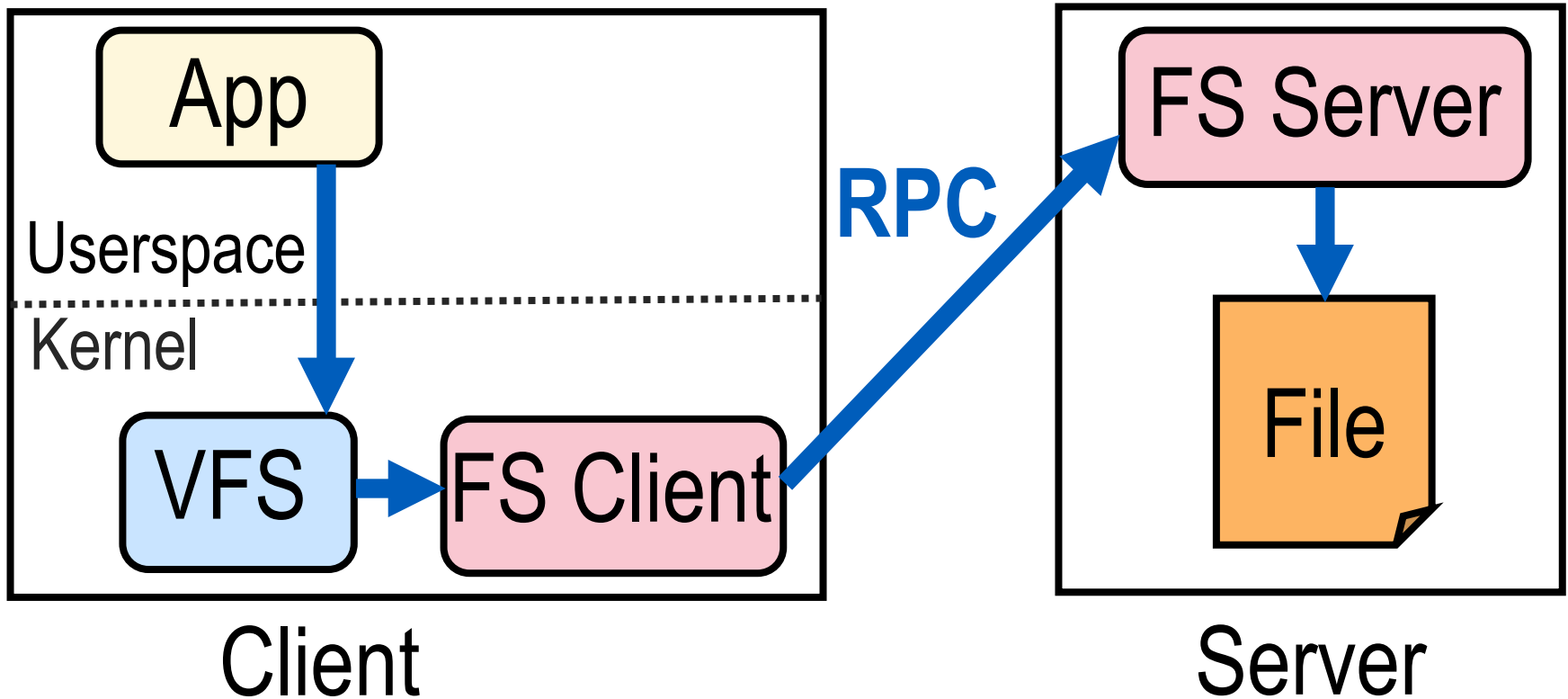




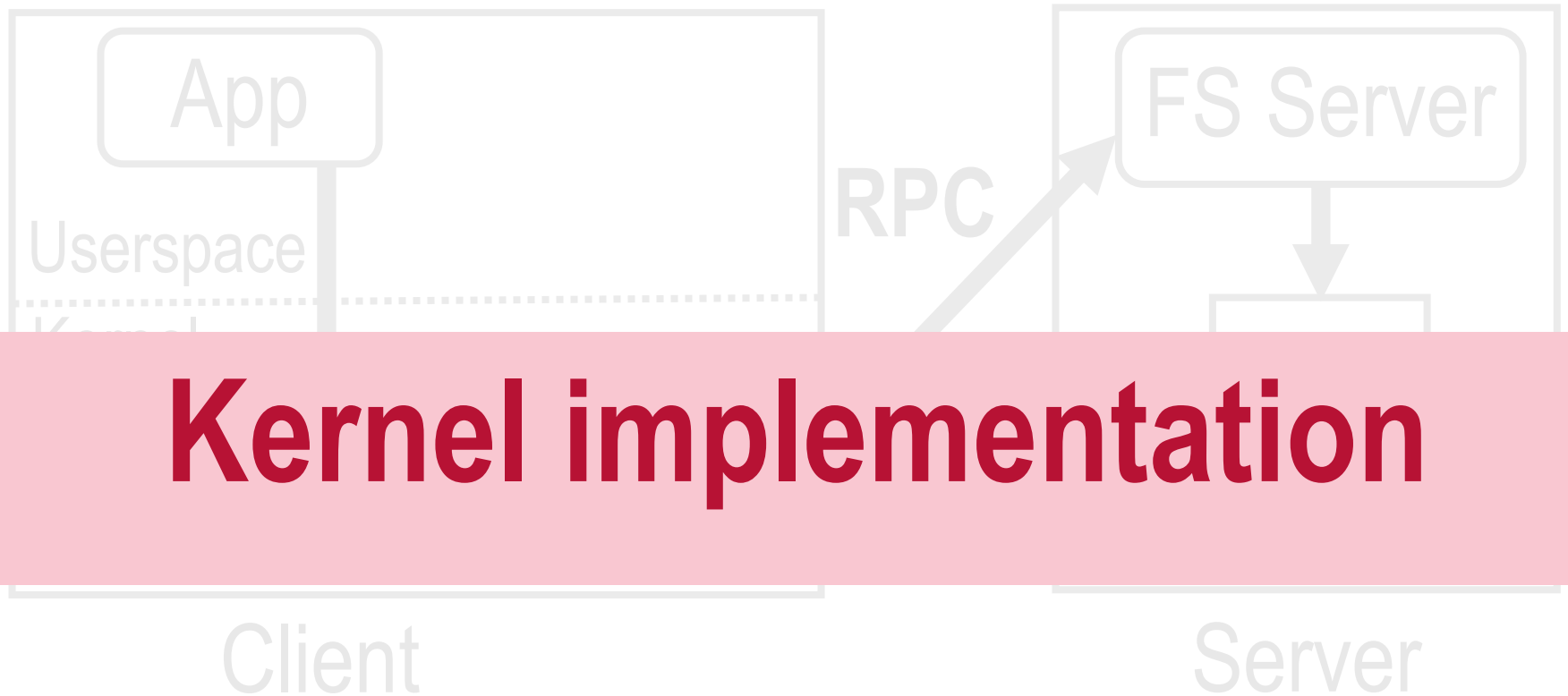
# Distributed File Systems



# Distributed File Systems

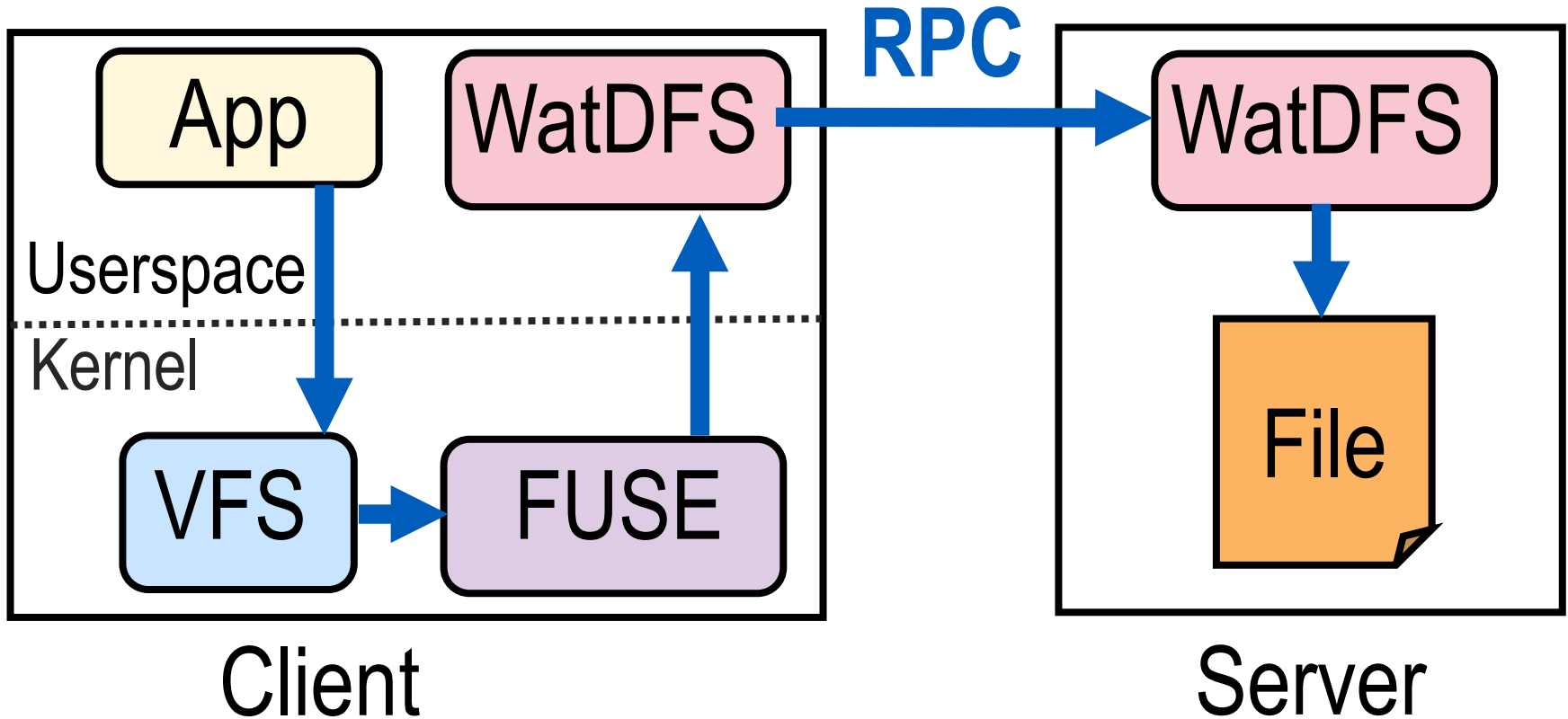


# Distributed File Systems





# Distributed File Systems



# Distributed File Systems

**Provides access  
transparency**

**Userspace implementation**

# WatDFS Project

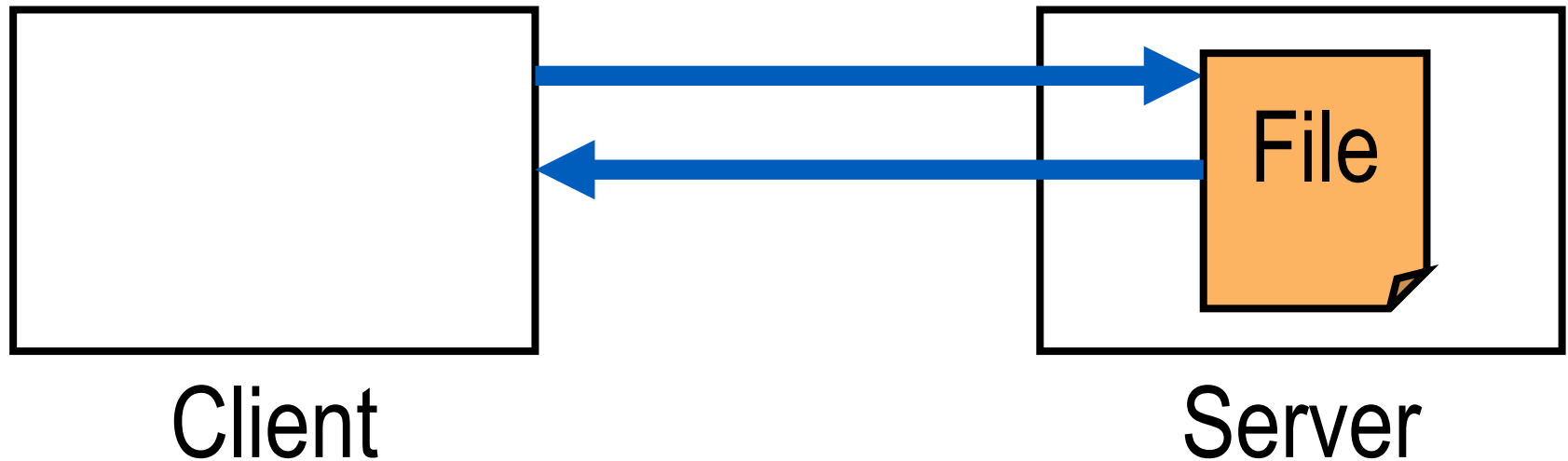
**Implement** WatDFS client and server:

**Support:** file creation, open, close, read, write, truncate, and metadata operations

Using two distributed file systems models

# Remote-Access Model

Forward operations to server



File stays at server

# Remote-Access Model

Forward operations to server

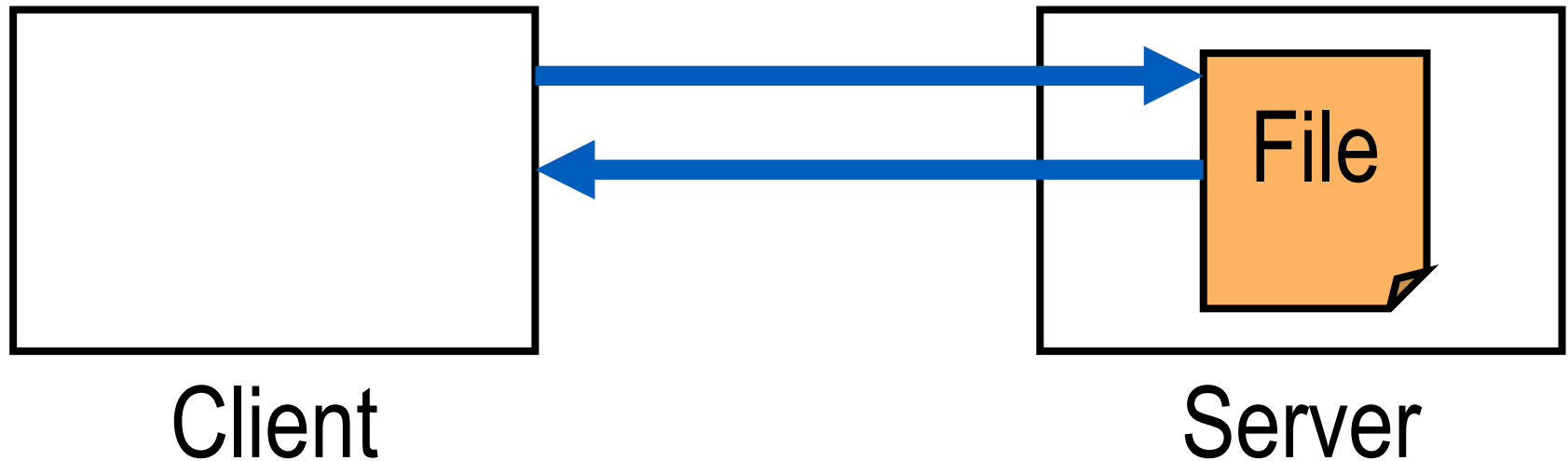


**Increases Latency**

File stays at server

# Remote-Access Model

## Learning Goals

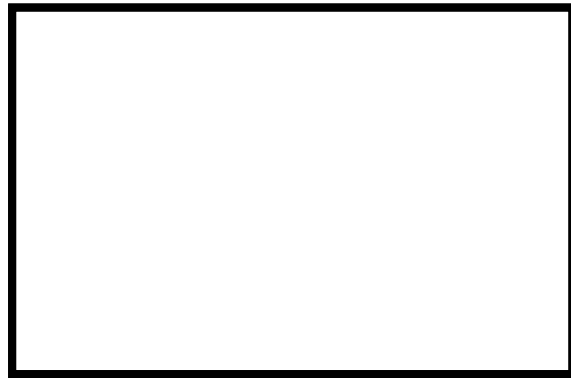


**Introduce** RPCs and file I/O

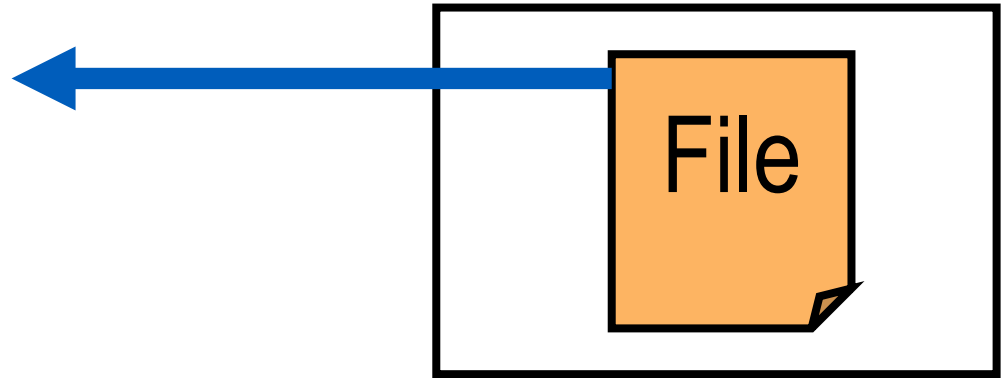
**Familiarize** tools (`libfuse`, `gdb`, `strace`)

# Upload-Download Model

**Download** file from server



Client



Server

# Upload-Download Model

**Download file from server**



**Perform  
operations  
at client**



# Upload-Download Model

**Download file from server**



Client  
Perform  
operations  
**at client**

# Upload-Download Model

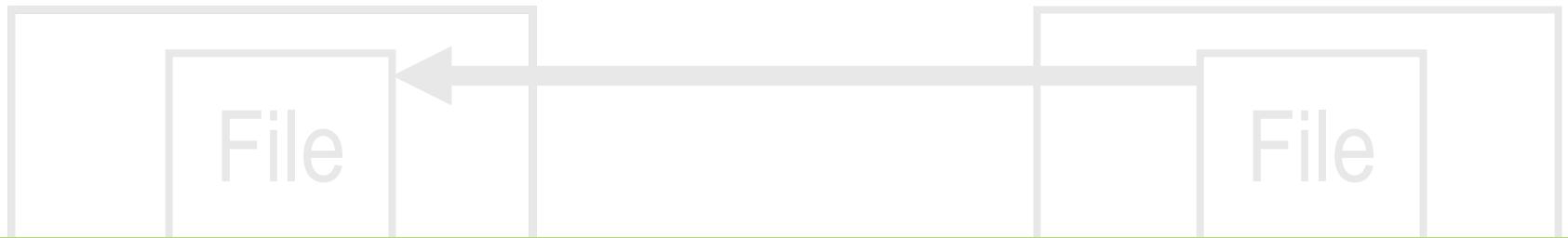
**Download file from server**



**Perform  
operations  
at client**

# Upload-Download Model

Download file from server

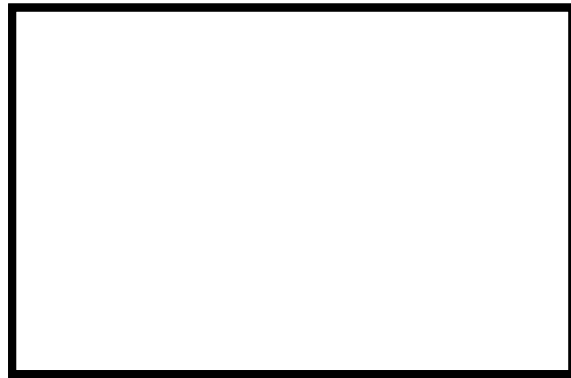


**Reduces Latency**

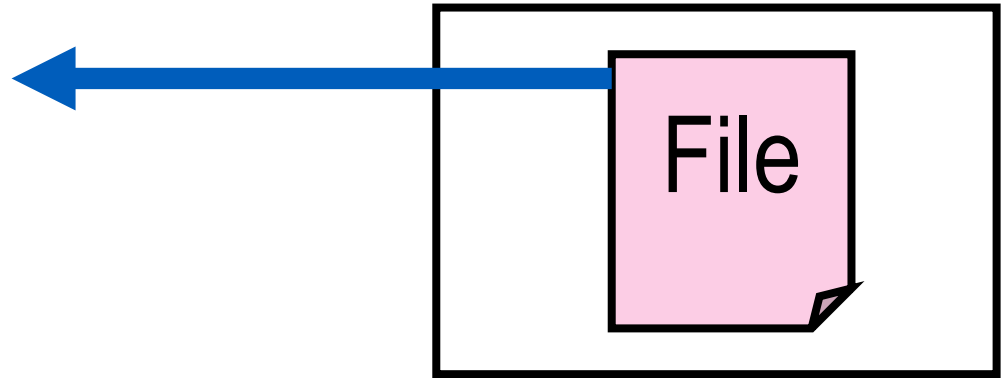
Perform  
operations  
at client

# Upload-Download Freshness

**Download** file from server



Client



Server

# Upload-Download Freshness

**Download file from server**



**Perform  
operations  
at client**

# Upload-Download Freshness

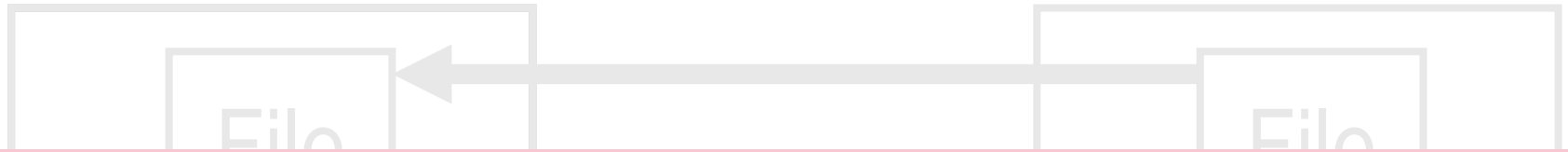
**Download file from server**



**Perform  
operations  
at client**

# Upload-Download Freshness

Download file from server

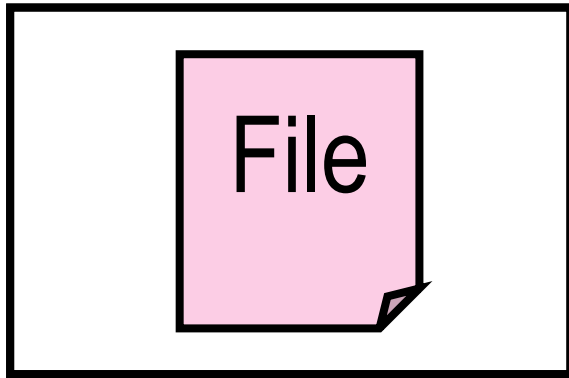


**Clients see stale state**

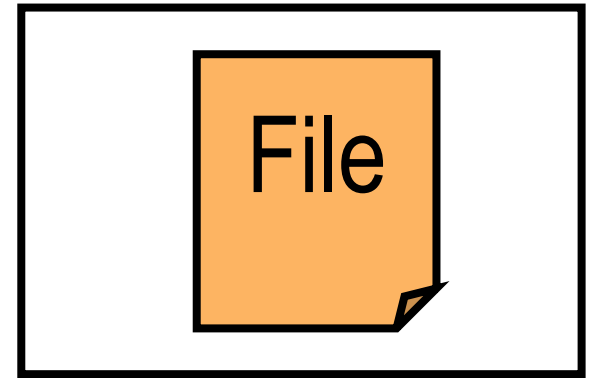
Client  
Perform  
operations  
at client

Server

# Upload-Download Freshness



Client



Server

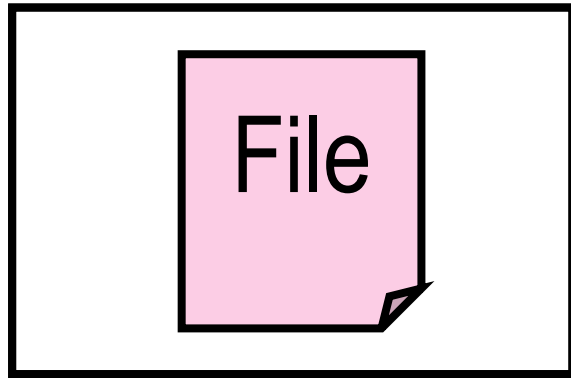
How to ensure **freshness**?

**Periodically** upload and download

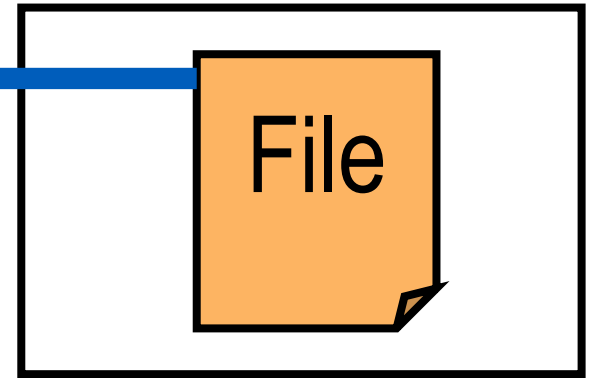


# Upload-Download Freshness

**Download** file from server



Read Client



Server

# Upload-Download Freshness

Download file from server



# Upload-Download Freshness

Download file from server



Perform  
operations  
**at client**

# Upload-Download Freshness

Download file from server



Read Client

Server

Perform  
operations  
**at client**

# Upload-Download Freshness

Download file from server



Read Client

file from server

Server

Perform  
operations  
at client

# Upload-Download Freshness

Download file from server



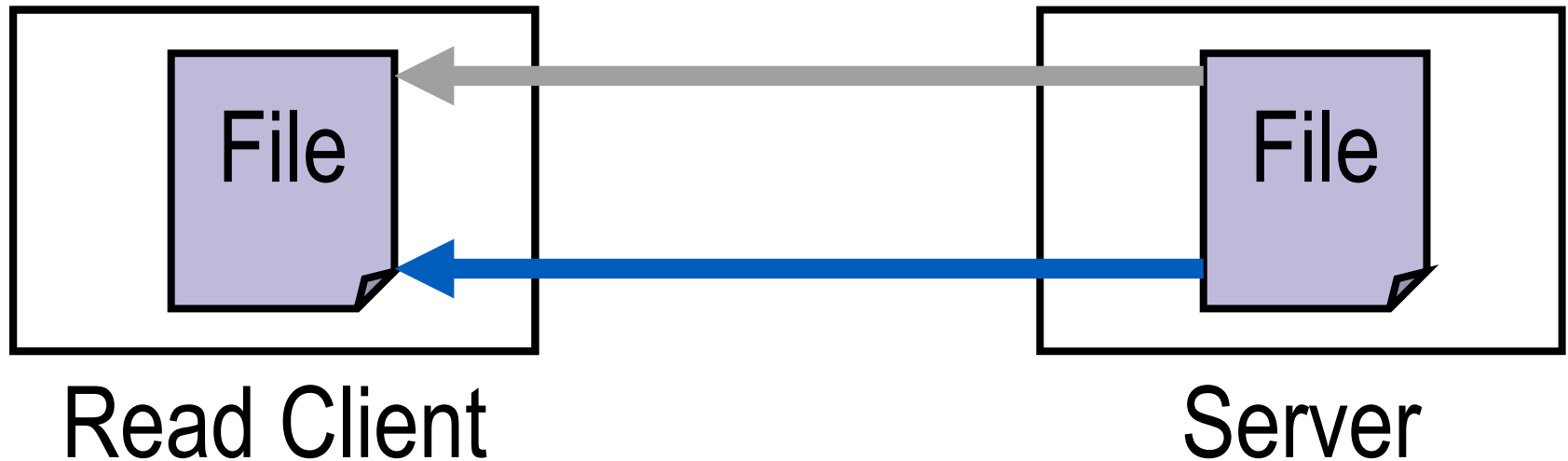
Read Client

file from server

Server

Perform  
operations  
at client

# Upload-Download Freshness



Periodically upload and download using  
**timestamp-based cache consistency**

# Upload-Download Freshness

**Clients see fresher state**

**Reduces Latency**

**Details at [tiny.cc/watdfs](https://tiny.cc/watdfs)**



# Upload-Download Model

## Learning Goals



**Manage** distributed state with cache consistency

**Use** locks for atomicity and mutual exclusion

# Experiences with WatDFS

## Provide students with:

Detailed specification and Q&A forum

Public and release **tests** (Marmoset)

Starter code: ~300 lines of code

# Experiences with WatDFS

## Students implement

~760 lines of code for remote-access model

~1425 lines of code for upload-download model

**Design** document about upload-download model

# Experiences with WatDFS

95% **passed all** remote-access model tests

80% **passed majority** of  
upload-download model tests

Most **common errors and questions** about  
timestamp-based cache consistency

# WatDFS Project Summary

**Implement** two distributed file systems models

Covers wide range of course material

FUSE allows **usage of existing applications**

Tests provide **high-quality & timely feedback**

**Details at [tiny.cc/watdfs](https://tiny.cc/watdfs)**