Course Overview and Introduction

Tyler Bletsch
Duke University

Slides include material from Vince Freeh (NCSU)
Instructor

• Professor: Tyler Bletsch
  • Office: Hudson Hall 106
  • Email: Tyler.Bletsch@duke.edu
  • Office Hours: See course site
MOTIVATION
Average person’s view of storage

storage = computar does it !!
Average engineer’s view of storage

storage = thing inside computer!!
A few enterprise storage architectures (1)

A few enterprise storage architectures (2)

- From: http://wiki.abiquo.com/display/ABI20/Monolithic+Architecture
A few enterprise storage architectures (3)

A few enterprise storage architectures (4)

Technical_Reference_Guide-Storage_Architecture.html
Why do all this? What problems are we solving?

- **Cost**: Is it cheap enough?
- **Capacity**: Can it hold enough?
- **Performance**: Is it fast enough?
- **Accessibility**: Can the data be accessed by everyone who needs it?
- **Security**: Is data protected from unauthorized access?
- **Reliability**: Is the downtime probability low enough?
- **Integrity**: Is data protected from hardware failures, disasters, and malicious attacks?
- **Compliance**: Do I keep data long enough safely?
- **Accountability**: Can I track all changes?
- **Space efficiency**: How much floor space do I need?
- **Power efficiency**: How many watts do I burn?
Why do all this? What problems are we solving?

- **Cost**: Is it cheap enough?
- **Capacity**: Can it hold enough?
- **Performance**: Is it fast enough?
- **Accessibility**: Can the data be accessed by everyone who needs it?
- **Security**: Is data protected from unauthorized access?
- **Reliability**: Is the downtime probability low enough?
- **Integrity**: Is data protected from hardware failures, disasters, and malicious attacks?
- **Compliance**: Do I keep data long enough safely?
- **Accountability**: Can I track all changes?
- **Space efficiency**: How much floor space do I need?
- **Power efficiency**: How many watts do I burn?

Color code: how well can a simple drive in a laptop let you control these variables?
Online course Info

- **Course Web Page:** static info
  - [https://people.duke.edu/~tkb13/courses/ece566/](https://people.duke.edu/~tkb13/courses/ece566/)
  - Syllabus, schedule, slides, assignments, rules/policies, prof/TA info, office hour info
  - Links to useful resources

- **Piazza:** questions/answers
  - Post all of your questions here
  - Questions must be “public” unless good reason otherwise
  - **No code** in public posts!

- **GradeScope:** Submit annotated PDFs for grading

- **Sakai:** just assignment submission and gradebook
Where to get info

• This info is fairly industry-connected, no great textbook
  • Semi-exception: “Evolution of the Storage Brain” by Larry Freeman (not a required text)

• Course material will come from lectures and supplementary readings
  • See course site for resources

• Additional independent research on your part will likely be necessary!
Stuff that’s different because we’re hybrid during COVID-19
Planning for failure or planning for success?

- How to definitely be miserable, depressed, and fail

  **Passive Watching**

  welp time to “attend” my “class” 😞
  ughhhhhhhhhhhhhhhhhhhhhhhhh

  (Later)
  oh no how do you computer?!??

- How to be engaged, interested, and pass

  **Active Engagement!**

  I have a question that I’m asking out loud in real time with real people, thus engaging my brain, getting valuable socialization, establishing a rapport with my peers!

  Also I have a beard now!

  (Later)
  im fine here too
The course is **synchronous**
  - **Video on please:** You were going to wear clothes and sit upright if this were in person, why change that?

Live attendance of **lecture** is expected
  - I may add quizzes if this doesn’t appear to be working

Live attendance of **special out-of-class labs** is expected
  - Only in the first few weeks – build your server!

Recordings will be made available for reference later
  (They will always be private to students in the course)

What if you have **real life reasons** why you can’t do the above?
  - You may request to take the course **asynchronously** by emailing me directly.
  - Valid reasons: crazy timezone, complicated living/working situation
  - Invalid reason: preference
CRITICALLY IMPORTANT TO
GOOD HAPPY SUCCESS

Interrupt me with

QUESTIONS!!!!!!
1. Try the “Raise hand” button in Zoom or in real life

2. If I don’t notice in 10 seconds, **unmute and make sounds until I stop.**

“It’s not rude if I’m literally telling you to do it.”
These are extraordinary times.

The course conditions are extraordinary.

The Duke response has been extraordinary.

The logistics of running a course like this hybrid will be extraordinary.

Let our commitment to each another and to ourselves also be extraordinary!
## Grading Breakdown

<table>
<thead>
<tr>
<th>Category</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project proposal draft</td>
<td>2%</td>
</tr>
<tr>
<td>Project proposal</td>
<td>3%</td>
</tr>
<tr>
<td>Project milestone 1</td>
<td>5%</td>
</tr>
<tr>
<td>Project milestone 2</td>
<td>5%</td>
</tr>
<tr>
<td>Project final report</td>
<td>5%</td>
</tr>
<tr>
<td>Project final presentation</td>
<td>5%</td>
</tr>
<tr>
<td>Project final demo</td>
<td>15%</td>
</tr>
<tr>
<td>Homeworks/programs/labs</td>
<td>40%</td>
</tr>
<tr>
<td>Midterm exam</td>
<td>10%</td>
</tr>
<tr>
<td>Final exam</td>
<td>10%</td>
</tr>
</tbody>
</table>
HOMEWORKS, LABS, AND PROGRAMS
Lab Motivation: What is a computer?

- Computers are:
  - Abstract theoretical math engines that float around on the internet?
  - **NO!**

- **PHYSICAL OBJECTS**
- **MADE OF MATERIALS**
- **IN THE REAL WORLD**
- **AND YOU CAN TOUCH THEM**
- **AND PUT THEM PLACES**
- **WITH YOUR ARMS/LEGS/FINGERS/BODY**
- **AND LIKE A SCREWDRIVER OR WHATEVER!!!!!!!**
Result: this course is HANDS ON

- Historically, the most popular assignments have been the realistic, hands-on ones. So I’ve added a lot of hands-on experience to the course.

- Each student group will be assigned a physical storage server which is upstairs in Hudson 214

- **Lab 0** will have you prepare and deploy this server.

- **Labs 1+** will have you do realistic storage tasks on it.
Labs
• Group work
• Hands-on
• Usually on your server
• Submitted via GradeScope (and Sakai for code)
• Can discuss concepts with other groups, but not answers

Homeworks
• Individual work
• Pen-and-paper questions
• Submitted via GradeScope (and Sakai for code)
• Can discuss concepts with others, but not answers
Also: a few “Program” assignments

• Programs will involve writing system code using BUSE (Block device in USErspace) and FUSE (Filesystem in USErspace)

• Assignments “Program 0” (BUSE) and “Program 1” (FUSE) are individual
  • Prepare each individual for the late-semester group project
Late penalties

• Late homework/lab/program incur penalties as follows:
  • Submission is 0-24 hours late: total score is multiplied by 0.9
  • Submission is 24-48 hours late: total score is multiplied by 0.8
  • Submission is more than 48 hours late: total score is multiplied by the Planck constant (in J·s)

• NOTE: If you feel in advance that you may need an extension, contact the instructor.
Class lab sessions to kickstart homework

• We’re going to schedule a few **class-wide lab sessions** so everyone can start to work on their server with instructor support
  • Why not a separate lab section? We don’t need every week...

• Be sure to respond to the **scheduling survey** that I sent; deadline is end of today!
You will eventually deploy your server in a real datacenter: the FitzWest server room in the CIEMAS basement.

- This means you’ll have **badge access** to a real datacenter.

Datacenter rules (you need to sign this to get access):

1. **Don’t touch other people’s stuff.** Includes other racks, other equipment, and other group’s servers in this course. You can touch your server, its cables, and shared tools.

2. **Respect shared resources.** The room has LCD monitors, keyboards, carts, screwdrivers, etc., which you can use. You must not interfere with IT operations and you must put stuff away when done.

3. **Report issues promptly.** Tell me if anything’s wrong.

< Sign online to gain access
- Half-semester effort in some area of storage
- Several choices (plus choose-your-own)
- Instructor feedback at each stage

- Any stage can result in a need for **resubmission** (grade withheld pending a second attempt).

- **See course site project page** for details
The Project

- **Draft proposal**: Say what you’re going to do and how.
  - Write-up plus 60-minute meeting scheduled out of class.
  - Must include weekly schedule and two milestones (demoed on workdays).
  - Get feedback

- **Final proposal**: Incorporate feedback from above.

- **Milestones and workdays**: Evidence of progress that you demo to me directly; also includes free time to collaborate.

- **Final report**: Describe and evaluate your work (max 8 pages).

- **Final presentation**: Demo your work and explain the implementation process to the class (15 min).

- **Final demo**: Defend your project to the instructor.
  - 90+ minute meeting scheduled out of class.

- Read course page for details!
But what *is* the project?

- Generally starts with your BUSE or FUSE program
- Create a system with feature(s) that improve one or more of:
  - Availability/recoverability
  - Network-accessibility
  - Storage efficiency
  - Performance
  - Security
- Alternately, you may propose a **wildcard project** (custom goal, may or may not use BUSE/FUSE at all)
Example projects

- **Availability/recoverability**
  - RAID at the filesystem level
  - Mirroring to second system (or cloud?)
- **Network-accessibility**
  - Make a network filesystem
  - Store to cloud service
- **Storage efficiency**
  - Filesystem deduplication
  - Filesystem compression
- **Performance**
  - Minimal-seek on disk data structures
  - Caching with read-ahead
  - Hybrid SSD+HDD filesystem
- **Security**
  - Access control list support
  - Per-user at-rest file encryption

**Wildcard projects**
- Special purpose file system (e.g. MP3 transcoding)
- Custom block device instead of file system:
  - Custom RAID
  - Custom SAN
  - Block-level encryption
  - Block-level compression
  - Block-level deduplication
Be thinking about possible projects as we go!

We’ll revisit project selection closer to the proposal...
POLICIES
Grade Appeals

• All regrade requests must be in writing
  • Email the TA who graded the question
    (we’ll indicate who graded what)

• After speaking with the TA, if you still have concerns, contact the instructor

• All regrade requests must be submitted no later than 1 week after the assignment was returned to you.
Academic Misconduct

• Academic Misconduct
  • Refer to Duke Community Standard
  • Labs are groupividual – everyone works on it
  • Common examples of cheating:
    • Running out of time and using someone else's output
    • Borrowing code from someone who took course before
    • Using solutions found on the Web
    • Having a friend help you to debug your program

• **I will not tolerate any academic misconduct!**
  • Software for detecting cheating is very, very good ... and I use it

• “But I didn’t know that was cheating” is not a valid excuse
Our Responsibilities

• The instructor and TA will...
  • Provide lectures/recitations at the stated times
  • Set clear policies on grading
  • Provide timely feedback on assignments
  • Be available out of class to provide reasonable assistance
  • Respond to comments or complaints about the instruction provided

• Students are expected to...
  • Receive lectures/recitations at the stated times
  • Turn in assignments on time
  • Seek out of class assistance in a timely manner if needed
  • Provide frank comments about the instruction or grading as soon as possible if there are issues
  • Assist each other *within the bounds of academic integrity*
Course summary

- We have **hard disks** and **solid-state drives (SSDs)**
- We can use **RAID** to combine performance and capacity while masking effects of drive failure
- The concept of files and directories comes from **File Systems**, a rich field of study.
- We can provide virtual disks to users over **Storage Area Network (SAN)** protocols
- We can provide file access to users using **Network-Attached Storage (NAS)** protocols
- We can provide **storage as a service (SaaS)** via cloud-type protocols.
- Storage efficiency can be improved with **data deduplication** and **compression**.
- We need to preserve **business continuity**: avoid downtime and lost data through **backups** and **high availability**
- Storage arrays are deployed based on **workload sizing**.
- Storage is often folded into a complete hardware/software stack: **converged architecture**.
- Storage systems are large enough that **management/monitoring** is its own challenge.
- Storage architects need to understand **basic finance** and **legal/compliance issues**
INTRODUCTION TO STORAGE DEVICES
Basic storage device history

- From https://aaronlimmv.wordpress.com/2013/05/02/types-of-storage-and-basic-advantages-and-disadvantages/
The ancient model of large enterprise storage

- **DASD: Direct Access Storage Device**
  - Starting with the IBM 350 in 1956
  - Your One Big Computer accesses your One Big Drive
  - Evolution: make the One Big Drive bigger and more reliable
  - Result: The One Big Drive became more and more expensive and critical
  - **Problem?**
DASD problem: single point of failure

- The DASD was a single point of failure with *all* your data
  - Better treat it gently...

Man with amazing fashion sense moves a 250MB disk, circa 1979.
Key trend: consumerization

- A common evolution in IT:
  - Businesses use a fancy expensive “Enterprise Thing”.
  - Normal people get a cheaper version, “Consumer Thing”. It’s cheap and good enough.
  - Consumer Thing gets better and better every year because:
    - There are more consumers than businesses (bigger market)
    - There are more vendors for consumers than for businesses (more competition)
    - The margins are thinner for consumer goods (more cut-throat competition)
  - A Smart Person finds a way to use the Consumer Thing for business.
  - Industry experts call the Smart Person dumb and say that no real business could ever use the Consumer Thing.
  - The Smart Person is immensely successful, and all businesses use the Consumer Thing.
  - Industry experts pretend they knew all along.
Consumerization in servers

- Big business use mainframe computers
- Everyone else uses microcomputers
- Microcomputers beat mainframes
- We start calling them “servers”
- Mainframes almost entirely gone
• Big business use DASDs

• Everyone else eventually gets small hard disks (SCSI)

• Disk arrays invented using “JBOD” and eventually “RAID”

• Storage companies based on disk arrays gain traction

• DASDs are entirely gone
Disk arrays

- **JBOD**: Just a Bunch Of Disks
  - Multiple physical disks in an external cabinet
  - Array is connected to one server only.
  - Provides higher storage capacity with increased number of drives.
  - Effect on performance?
  - Effect on reliability?

- Can we do better?
Disk arrays

- **RAID**: Redundant Array of Inexpensive Disks
  - Academic paper from 1988
  - Revolutionized storage
  - Will discuss in depth later
  - Combine disks in such a way that:
    - Performance is additive
    - Capacity is additive
    - Drive failures can occur without data loss
  - Still directly attached to one server
Next step: intelligent arrays

- Server acts as host for storage, provides access to other servers
  - Dedicated hardware for RAID
  - Optimized for IO performance
  - High speed cache
  - Can add various special features at this layer: access controls, multiple protocols, data compression and deduplication, etc.
Method of Attachment

- How to connect storage array to other systems?
  - DAS: Direct Attached Storage
    - One client, one storage server
  - SAN: Storage Area Network
    - Storage system divides storage into “virtual block devices”
    - Clients make “read block”/“write block” requests just like to a hard drive, but they go to the storage server
  - NAS: Network-Attached Storage
    - Storage system runs a file system to create abstraction of files/directories
    - Clients make open/close/read/write requests just like to the OS’s local file system
DAS: Direct Attached Storage

- One-to-one connection
- Historically: connect via SCSI ("Small Computer Systems Interface")
  - Even though actual SCSI cables/drives/systems are gone, the software protocol is still *everywhere* in storage. We’ll see it again very soon*.

- Modern:
  - USB: External drives, very fast as of USB 3.0
  - SATA (or if it’s external, e-SATA): The protocol modern consumer drives use
  - SAS (Serial Attached SCSI): The protocol modern enterprise drives use
  - NVMe (put SSD right on PCIe bus): for card-type SSDs

* see, I told you.
SAN: Storage Area Network (1)

- Split the aggregated storage into virtual drives called Logical Units (LUNs)
- Clients make read/write requests for blocks of “their” drive(s)
- Storage server translates request for block 50 of client 2 to actual block 4000
  (which in turn is block 1000 of disk 3 of the RAID array)
SAN: Storage Area Network (2)

- **Historical protocol: Fibre Channel (FC)**
  - A special physical network just for storage
  - Totally unlike Ethernet in almost every way
  - Still popular with very conservative enterprises
  - Actual traffic is SCSI frames
  - Clients and servers have special cards: a Host Bus Adapter (HBA) for FC

- **Modern protocols:**
  - Fibre Channel over Ethernet (FCoE):
    - Requires FCoE-capable switch
    - SCSI inside of an FC frame inside of an Ethernet frame
    - Clients and servers have special cards: a Converged Network Adapter for FCoE/Ethernet
  - iSCSI:
    - SCSI inside of an IP frame, usually inside of an Ethernet frame (but it’s IP, so it could be inside a bongo drum frame)
    - No special switch or cards needed (though iSCSI HBAs do technically exist)
NAS: Network-Attached Storage (1)

• Put a file system on the storage server so it has the concept of files and directories

• Clients make open/close/read/write requests for files on the remote file system
NAS: Network-Attached Storage (2)

- No special network or cards – works on normal IP/Ethernet
- Network File System (NFS):
  - Common for UNIX-style systems, invented by Sun in 1984
  - Literally just turns the system calls open/close/read/write/etc into “remote procedure calls” (RPCs)
  - Many revisions, we’re up to NFS v4 now
- Server Message Block (SMB) also known as Common Internet File System (CIFS)
  - Microsoft Windows standard for network file sharing, developed around 1990
  - Really badly named
  - Many revisions, we’re up to SMB 3.1.1 now
  - Native on Windows, supported on Linux with Samba (client and server)
How to tell NAS and SAN apart

**NAS** = File

**SAN** = Block
System constraints

• What is a tradeoff?
• Constraints:
  • Cost
  • Physical environment
  • Maintenance & support
  • Compliance (regulatory/legal)
  • HW & SW infrastructure
  • Interoperability/compatibility
Management activities

- **Provisioning**: allocate storage based on requirements:
  - **Capacity** – capacity planning
  - **Performance** – workload profiling
  - **Security** – access rule creation, encryption policy
  - **Reliability** – type of redundancy, backup policy
  - **Other** – archival duration, regulatory compliance, etc.

- **Monitoring**: ensure proper functioning over time
  - **Capacity**: watch usage over time, identify workloads at risk of running out
  - **Performance**: collect metrics at storage layer and/or application layer, compare to requirement, alert on violation/deviation, add resources as needed
  - **Security**: verify access control rules, deploy intrusion/anomaly detection, ensure at-rest and in-flight encryption is used where appropriate
  - **Reliability**: receive alerts when failures occur at any layer, continually ensure that availability and backup policies remain satisfied

- **Archival/destruction**: retire data properly
The data lifecycle

Questions?
FUSE

- **File System in Userspace**: Write a file system like you would a normal program.
- **You** implement the system calls: open, close, read, write, etc.

Figure from Wikipedia: http://en.wikipedia.org/wiki/Filesystem_in_Userspace
FUSE Hello World

```bash
~/fuse/example$ mkdir /tmp/fuse
~/fuse/example$ ./hello /tmp/fuse
~/fuse/example$ ls -l /tmp/fuse
total 0
-r--r--r-- 1 root root 13 Jan 1 1970 hello
~/fuse/example$ cat /tmp/fuse/hello
Hello World!
~/fuse/example$ fusermount -u /tmp/fuse
~/fuse/example$
```

- Let’s walk through it:
  [https://github.com/libfuse/libfuse/blob/master/example/hello.c](https://github.com/libfuse/libfuse/blob/master/example/hello.c)