

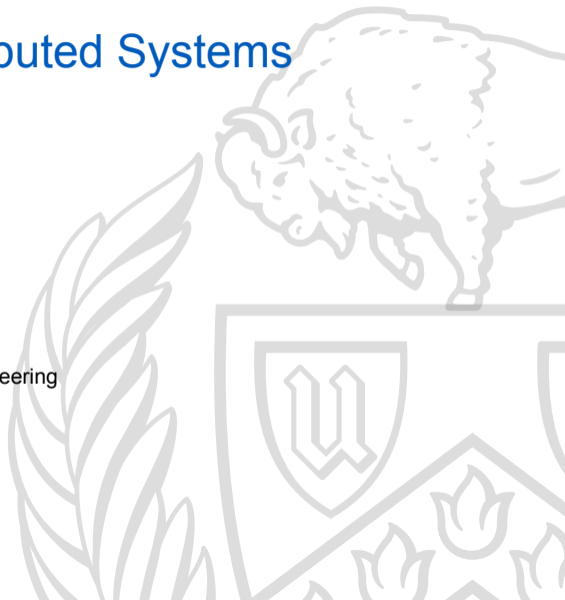
CSE 486/586: Distributed Systems

Introduction

Ethan Blanton

Department of Computer Science and Engineering

University at Buffalo



Welcome to CSE 486/586

My name is Ethan Blanton

Contacting me:

Email: `eblanton@buffalo.edu`

Office: Zoom

Office Hours: Mondays 10:00–11:00
Thursdays 14:00–16:00

The syllabus [1] is under References.

All slides are [on the web page](#), OH info is on Piazza.

Distributed Systems

Distributed systems are:

- multiple **computer programs**
- possibly spread out over **different networked components**
- **communicating** by **passing messages**

Many **modern software services** are distributed systems.

Distributed Systems are Hard

What makes them hard?

In a word: **failures**.

Computers fail, networks fail, software fails.

Failures can be **incidental** (e.g., power failures).

Failures can be **orchestrated** (e.g., a virus or worm).

Add to this challenges of **scale** and **concurrency**.

Failures

Large systems are **unreliable**:

Backblaze reported an **0.89% drive failure rate** in Q3 2020 [6].

Facebook says “In general, we always expect some degree of hardware failure in our data centers” [7].

Microsoft Azure reports that “Last year [written Dec 2020] alone, there were hundreds of routing outages or incidents, such as route hijacking and leaks.” [4]

Scale

Distributed systems can be **mind-bogglingly huge**:

YouTube serves **over a billion hours a day**. [10].

The Facebook Scuba Trailer service processes more than **180 GB per second** of streaming data [8].

Slack produces **100s of Terabytes of observability data** (debugging information) per day [5].

Concurrency

All of this happens **concurrently**:

Over **500 hours of video are uploaded every minute** to YouTube servers [10].

The Facebook Scuba Trailer runs on **over 2,000 machines** with a total of **over 96,000 cores** [8].

Expectations

For this course, I expect that you:

- Will be **respectful** to me, TAs, classmates
- Attend (or view, if attendance is impossible) **every lecture**
- **Adhere strictly** to the academic integrity policy
- Will seek assistance **early** if necessary
- Meet prereqs; among other things:
 - Have significant experience programming
 - Have a basic understanding of OS interfaces and networking
 - Can rapidly learn (or already know) Go

Most of all, **behave as adults** and strive to **maximize** your and your classmates' **learning experience** in this course.

Communication

We will communicate via:

- **Piazza** (our main tool; you should receive an invite)
- UB Email ([check yours regularly](#))
- YouTube Live (lectures, some open office hours)
- Zoom (office hours)
- UBlearns (gradebook, quizzes, tests, Panopto videos)

Ensure that you have access to [all of these things](#).

Having a Pleasant Semester

I intend for this course to be **fun and rewarding**.

You'll get out of it **what you put in**; no more, no less.

I do not take well to grade negotiation.

If you want a **better grade**, do **better work**.

If you're willing to put in the time, **I'm willing to help**.

TL;DR

If you cheat, you will fail.

International students: this is not just something I say.

Basic Policies

You must be familiar with the official policies. [9, 2, 3, 1]

All assignments and exams in CSE 486/586 **are individual.**

Collaboration with other students is forbidden **unless explicitly stated otherwise.**

Only course-supplied resources are allowed (more later).

Interactions with Other Students

Examples of interactions with other students that **are allowed**:

- Asking questions in open office hours, listening to answers
- Discussing **course concepts** (such as the course text)

Examples that are **not allowed**:

- Discussing implementation strategies with other students
- **Whiteboarding or writing pseudocode** with another student
- Looking at another student's screen
- Sharing code, GitHub repositories, *etc.* with another student

Sanctions

The default sanction in this course is **failure in the course**.

If you are found to have committed **any AI infraction**, **you will fail**.
Things that are not excuses:

- Ignorance
- Good intentions
- Past performance
- Accident

Sanctions are processed through the Office of Academic Integrity and become part of your University record.

Amnesty Policy

If you commit an AI violation and wish to retract it, **you can**.

However, it must be retracted **before I notice it**.

Retracted assignments:

- Receive a **zero in the gradebook**
- Are **not processed as an AI violation**

Retractions must be **in writing** and **explain the violation**.

Allowable Resources

The **only resources** that are allowed for this course are:

- The lecture slides I provide
- Material covered in lecture
- Required or recommended readings from lecture
- Documentation on golang.org

Resources other than these are **not allowable**, and use of other resources **may result in AI proceedings**.

Example Problematic Resources

Examples of resources that may land you in hot water:

- [Stack Overflow](#)
- [GitHub](#)
- [Geeks for Geeks](#)
- [Chegg](#)
- [Course Hero](#)
- Your work from previous semesters

This is not an exhaustive list!

Asking is Always OK

It is **always OK** to ask whether a resource is allowable.

It is **always OK** to ask course staff for assistance.

When in doubt, **ask**.

Programming Projects

A significant portion of your course grade will be projects.

- They are **individual projects**.
- Projects must be implemented in **Go**.
- You will receive detailed specifications.
- You will have to **do your own testing**.

Git and GitHub Classroom

We will use Git, GitHub, and GitHub Classroom.

You **must have** (or create!) a GitHub account.

You are **expected to use git and GitHub** for development.

We will **only look at your code on GitHub** for assistance.

Git help: Git book [!\[\]\(758ebdf4629c903da74c2e079717ae32_img.jpg\)](#), tutorial [!\[\]\(e7d82ae1e31b23b67694dcc1e3031ff6_img.jpg\)](#), Google [!\[\]\(e4aa5dd07782217adf10903e7f7dc845_img.jpg\)](#)

Project Assistance

Your TAs will be your primary source of project help.

To get the most out of your TAs, **do**:

- try the obvious things first,
- create minimal examples to show problems, and
- **consult the documentation.**

To avoid wasting TA time and failing to get help, **don't**:

- ask for help before you've tried to understand the problem
- **start at the last minute.**

Project Submission

We will submit using Autograder.

Submission rules:

- Submitted w/in 24 hours of the deadline: -20%
 - Doesn't count Saturday or Sunday
 - Doesn't count University holidays
- Projects submitted after 24 hours will not be accepted

Example 1: Project is due Friday at 11:59 PM, turned in Monday at 3 PM — 20% penalty.

Example 2: Project is due Monday at 11:59 PM, turned in Wednesday at 12:15 AM — not accepted.

Today's Assignments

As soon as possible:

- Read the syllabus [1]
- Watch the AI video on Panopto (via UBlearns)

By **Friday, February 12**:

- Complete the AI quiz on UBlearns
- Complete Programming Assignment 0 and turn it in
Make sure to work on it **before add-drop!**

Next Time ...

Some background on:

- Internet architecture
- Network communication

References I

Required Readings

- [1] Ethan Blanton. *CSE 486/586 Syllabus*. Jan. 2021.
- [2] *Department of Computer Science and Engineering Academic Integrity Policy*.
<https://engineering.buffalo.edu/computer-science-engineering/information-for-students/policies/academic-integrity.html>.
- [3] *Ethan's Academic Integrity Policy*. https://cse.buffalo.edu/~eblanton/policy/academic_integrity/.
- [9] *University at Buffalo Academic Integrity Policy*.
<https://catalog.buffalo.edu/policies/integrity.html>.

References II

Optional Readings

- [4] [Albert Greenberg](https://azure.microsoft.com/en-us/blog/microsoft-introduces-steps-to-improve-internet-routing-security/). *Microsoft introduces steps to improve internet routing security*. <https://azure.microsoft.com/en-us/blog/microsoft-introduces-steps-to-improve-internet-routing-security/>. Dec. 2020.
- [5] [Suman Karumuri et al.](http://cidrdb.org/cidr2021/papers/cidr2021_abstract08.pdf) “Cloud Observability: A MELTING Pot for Petabytes of Heterogeneous Time Series”. In: *11th Annual Conference on Innovative Data Systems Research*. Jan. 2021. URL: http://cidrdb.org/cidr2021/papers/cidr2021_abstract08.pdf.

References III

- [6] Andy Klein. *Backblaze Hard Drive Stats Q3 2020*. <https://www.backblaze.com/blog/backblaze-hard-drive-stats-q3-2020/>. Oct. 2020.
- [7] Fred Lin, Harish Dattatraya Dixit, and Sriram Sankar. *How Facebook keeps its large-scale infrastructure hardware up and running*. <https://engineering.fb.com/2020/12/09/data-center-engineering/how-facebook-keeps-its-large-scale-infrastructure-hardware-up-and-running/>. Dec. 2020.
- [8] Yuan Mei et al. “Turbine: Facebook’s Service Management Platform for Stream Processing”. In: *IEEE International Conference on Data Engineering*. Apr. 2020.

References IV

- [10] *YouTube for Press, viewed 2021-01-18.*
[https://blog.youtube/press/.](https://blog.youtube/press/)

Copyright 2021 Ethan Blanton, All Rights Reserved.

Reproduction of this material without written consent of the author is prohibited.

To retrieve a copy of this material, or related materials, see <https://www.cse.buffalo.edu/~eblanton/>.