

# CPSC 416 – Distributed Systems

## Lecture 02 — Failure models, the network substrate, and evidence

UBC Vancouver

2026-01-08

## Recap: the course invariant

- ① Start from a failure (concrete, observable)
- ② State a guarantee (an invariant or conditional liveness)
- ③ Design a mechanism (protocol + state)
- ④ Produce falsifying evidence (what would prove you wrong)

# Today

- Failure models: what can go wrong (and what cannot be distinguished)
- Network service model: what the substrate does and does not promise
- Reliability is a policy: mechanisms you build above the transport
- Studio: GitHub repo + signing + evidence chain (make evidence operational)

## Today's failure: retry storm (vignette)

- October 2025 AWS us-east-1 outage *amplified* by retry storm
- A dependency outage disrupts name resolution or reachability (DNS/API endpoints).
- Clients time out and all retry, at once.
- Retrying traffic overloads recovery paths and prolongs the outage.
- Lesson: resilience includes *not making recovery harder*.

## Failure-first question

*What does the network promise you (and what does it explicitly not promise you)?*

## Network service model (best effort)

- Packets can be lost, delayed, duplicated, and reordered.
- Partitions happen: the network can fail without nodes crashing.
- TCP improves the local delivery contract, but does not solve distributed correctness.

# Failure models (naming what can go wrong)

- Fail-stop vs fail-slow vs Byzantine.
- Partitions vs crashes: what can be detected and what cannot?
- Your protocol is only correct relative to an explicit failure model.

# Why timeouts are not a detector

- A timeout can mean: crash, partition, congestion, fail-slow, GC pause, overload.
- Retries amplify failure: they add load and create duplicates.
- Correctness requires explicit policy: idempotency, ordering, deduplication, and backoff.

# Mechanisms cheat-sheet (early)

- Sequence numbers + dedup caches (handle duplication/reordering)
- Idempotency keys (make retries safe)
- Explicit acknowledgements (define “delivered”)
- Timeouts + backoff (control retry storms)
- Observability as a first-class mechanism (logs/metrics/traces as evidence)

## Mini-studio (5 min): name the failure model

- For the retry-storm vignette, write:
  - ▶ failure model (what happened)
  - ▶ guarantee (what must still be true)
  - ▶ falsifying evidence (what would prove you wrong)
- Share one “missing assumption” with a neighbor.

## Studio (30 min): GitHub evidence chain

- Goal: by the end, your repo has a signed commit on 'main' and a first verified ledger entry.
- Steps (high level):
  - ➊ Access your repo in the course GitHub org; clone it locally.
  - ➋ Configure commit signing (GPG or SSH) and verify GHES shows "Verified".
  - ➌ Install hooks ('githooks/') and run the evidence tooling once.
  - ➍ Commit + push; confirm branch protection blocks unsigned commits.

## Studio prompts (what to record)

- In 'ai/LOG.md': note decisions made today (signing method, tool setup) and what changed.
- In 'evidence/ledger.jsonl': append one entry for "Repo initialized + signing configured".
- In 'README.md' (optional): one sentence on how to run your code/tools locally.

# Logistics

- Readings + quiz (Week 01/02 on Canvas Readings page)
- Self-assessment opens at 13:00, due next class 11:30 (Pacific)
- DS Labs: keep ping scaffold unblocked