CPSC 416 Distributed Systems

Winter 2023 Term 1 (October 31, 2023)

Tony Mason (fsgeek@cs.ubc.ca), Lecturer

Logistics

Teaching Assistants

Andy Hsu (andy.hsu@alumni.ubc.ca)

Hamid Ramezanikebrya (hamid@ece.ubc.ca)

Jonas Tai (jonastai@student.ubc.ca)

Cathy Yang (kaiqiany@student.ubc.ca)

Office Hours

Remember: Use Piazza for **all** official course-related communications

- Not on Piazza? Not official.
- Canvas “comments/messages” **are not monitored**

Office Hours:

Who	When	Where
Tony	Monday 14:00-15:00 Wednesday 16:00-17:00	Discord
Andy	Thursday 19:00-20:30	Discord
Hamid	Friday 16:30-18:00	Kaiser 4075
Jonas	Thursday 13:00-14:00	X241
Cathy	Friday 09:00-10:30	X237

Self-Assessment

This week

- Usual self-assessment activity (Thu @ 17:00)
- DP2 Implementation Report Peer Feedback (Thu @ 17:00)

Next week

- Usual self-assessment activity (Tue/Thu @ 17:00)
- Capstone Week 3 Report (Tue @ 17:00)
- DP3 Implementation Code (Thu @ 17:00)
- DP3 Implementation Report (Thu @ 23:59)

Note:

- You are strongly encouraged to collaborate with others on this
- You should use tools at your disposal to answer these questions
- **Do not forget to submit it.**

Today’s Failure

Cloudflare

Date: October 4, 2023

Start: 07:00 UTC

End: 11:00 UTC

Duration: 4 hours

Impact: Domain name services experienced a 15% failure rate (compared to 3% normal)

Cause:

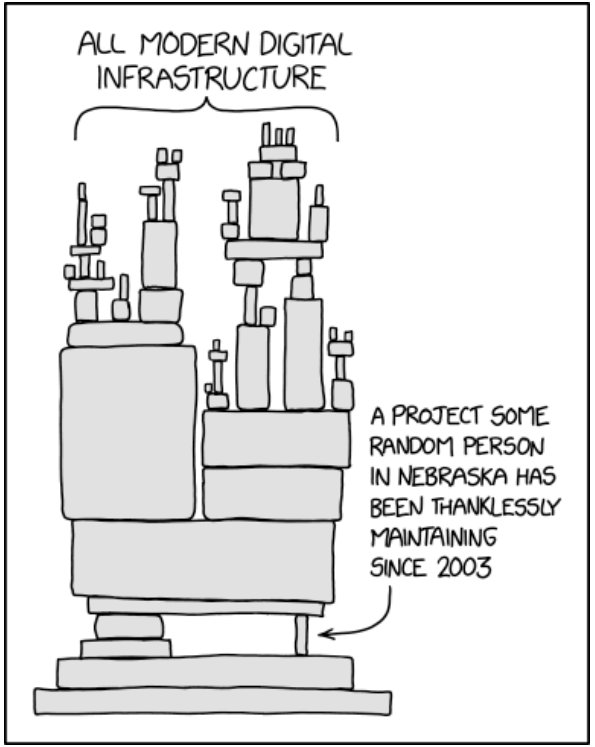
- A new DNS record type (ZONEMD) was added on 21 September
- Cloudflare automated fetch/parse of TLD zones began failing
- Cached versions began expiring on 4 October.
- DNSSEC validation began failing

Cloudflare

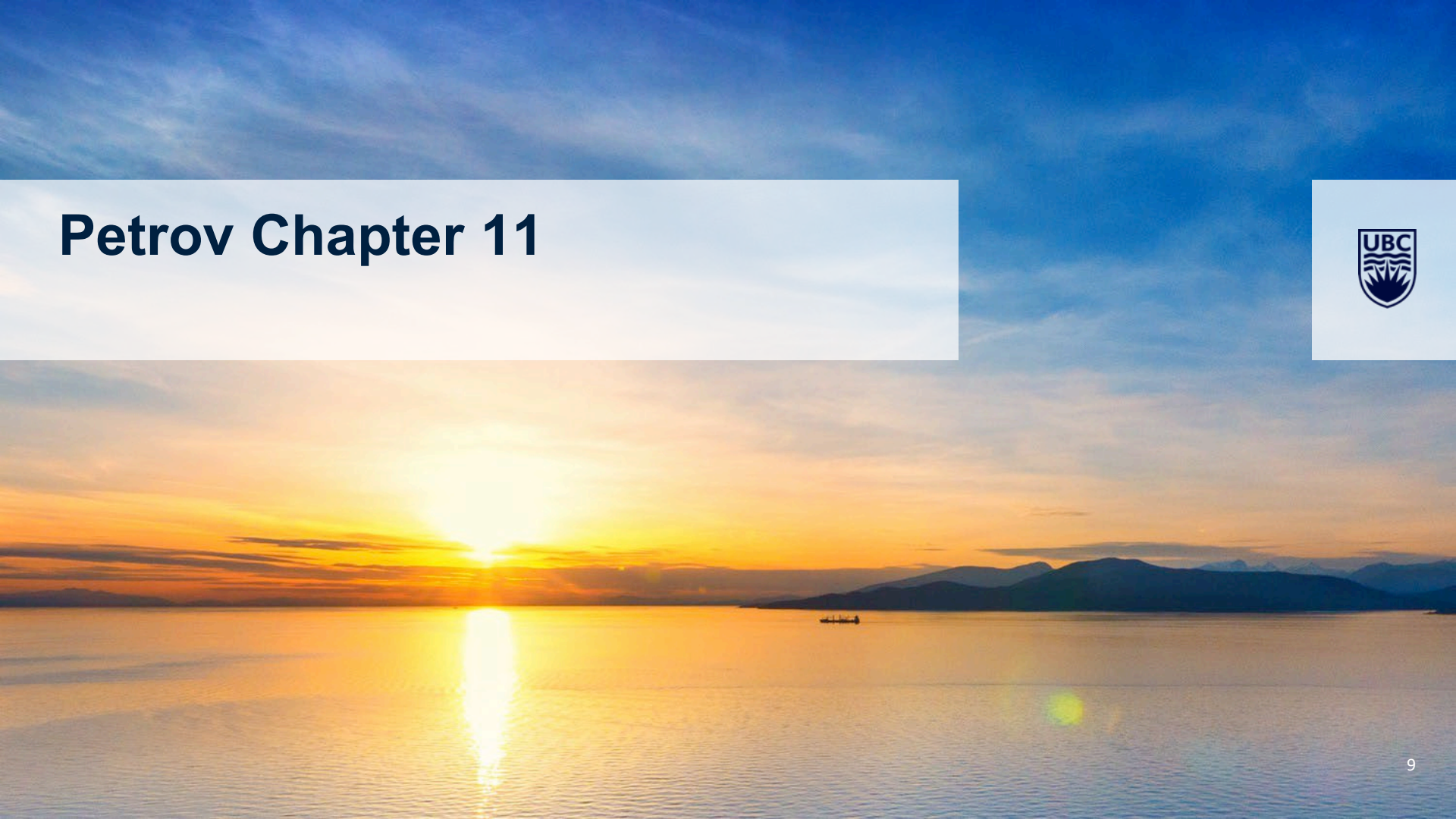
Solution: disable using cached zone file

- Note: temporary solution!
- Long-term fix is to parse the zone correctly

Take-away: many fragile components



Petrov Chapter 11



Learning Goals (Petrov Chapter 11)

Consistency Model Roles

Fault Tolerance via Redundancy

Atomic Updates

Consistency Models (redux)

Consistency Trade-offs

Multi-data centre replication

Why fault tolerant design is important

Strong Eventual Consistency

Understanding Consistency Models

Distributed systems consistency challenges:

- Network behaviour
- Consistency versus Availability trade-offs
- Replicated data convergence

Note that consensus protocols provide a *linearization*

- Agree on order of operations
- Data replication is *not* defined by the consensus protocol.

Consistency model defines requirements for data replication from client perspective

Replication for Fault Tolerance

Fault tolerance = “a property of a system that can continue operating correctly in the presence of failures of its components.”

Atomic update of data is challenging and expensive

- 100% quorum of replicas
- Atomic broadcast

Shared memory model

- Caching versus consistency (e.g., the “register” model)
  - Safe (returns an arbitrary value)
  - Regular (returns last completed write value)
  - Atomic (write is instantly visible everywhere upon completion)

Consistency Models (again)

Strict consistency = “complete replication transparency” or “atomic writes”

Linearizability = “write effect becomes visible to all readers at some point in time” for a *single object* – note that non-causally related changes are not ordered.

Sequential consistency = “ordering of operations are linearized for a single process”

Causal consistency = “ordering of operations are causally ordered.”

- Vector clocks provide a *partial order* (e.g., per node/process timestamps)

Session consistency = “client perspective ordering of operations”

Eventual consistency = “updates propagate asynchronously but are delivered over time”

Tunable consistency = “choosing how much failure can be tolerated by the system”

Strong Eventual Consistency

The “middle ground” between “eventual consistency” and “strong consistency”

- Allow divergence, with additional state to permit reconciliation

Conflict Free Replicated Data Types

- See Reddis
- Preclude conflict
- Do not require strict ordering
  - Side-effect free
  - Commutative
  - Causally ordered

Questions?


UBC
THE UNIVERSITY OF BRITISH COLUMBIA