CPSC 416 Distributed Systems

Winter 2023 Term 1 (September 28, 2023)

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

Logistics

Teaching Assistants

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

Hamidreza 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 11:00-12:30	X150, Table 1&2
Cathy	Friday 09:00-10:30 (Starting Sep. 22)	X237

Self-Assessment

Next week

- Post-lecture self-assessment activity – Due Tuesday (October 3 @ 17:00)
- Design Feedback – Due Tuesday (October 3 @ 17:00)
- Post-lecture self-assessment activity – Due Thursday (October 5 @ 17:00)
- Apparently, once the deadline passes you can't see the questions
  - Graded everything thus far – hope you can see it again

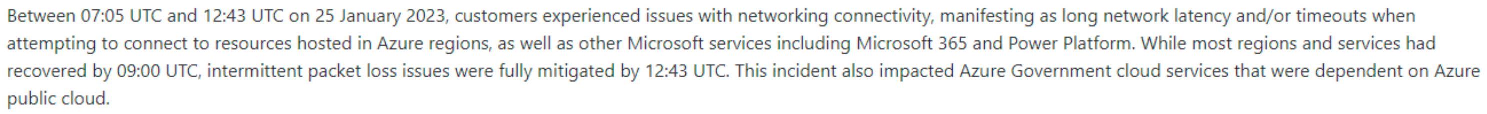
Note:

- 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.**
  - There are a couple students that haven't been submitting. ☹

Today’s Failure

Microsoft Azure Outage

January 25, 2023 07:05 UTC



Between 07:05 UTC and 12:43 UTC on 25 January 2023, customers experienced issues with networking connectivity, manifesting as long network latency and/or timeouts when attempting to connect to resources hosted in Azure regions, as well as other Microsoft services including Microsoft 365 and Power Platform. While most regions and services had recovered by 09:00 UTC, intermittent packet loss issues were fully mitigated by 12:43 UTC. This incident also impacted Azure Government cloud services that were dependent on Azure public cloud.

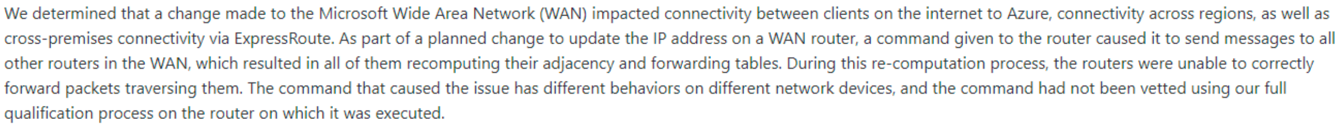
What does this *mean*?

- Microsoft 365 = “Office applications, including E-mail, OneDrive Storage, SharePoint, etc”
- Microsoft Power Platform = “data analytics”
- Azure Government = “Governmental services”

Microsoft Azure Outage

Why did it happen?

- Wide Area Network (WAN) reconfiguration event
- Changed IP address on a WAN router
  - Triggered messages *to* other WAN routers
  - Routers reconstructed adjacency lists & forwarding tables
  - During reconstruction *packet forwarding stopped*
- Change command “had not been vetted”



We determined that a change made to the Microsoft Wide Area Network (WAN) impacted connectivity between clients on the internet to Azure, connectivity across regions, as well as cross-premises connectivity via ExpressRoute. As part of a planned change to update the IP address on a WAN router, a command given to the router caused it to send messages to all other routers in the WAN, which resulted in all of them recomputing their adjacency and forwarding tables. During this re-computation process, the routers were unable to correctly forward packets traversing them. The command that caused the issue has different behaviors on different network devices, and the command had not been vetted using our full qualification process on the router on which it was executed.

Microsoft Azure Outage

January 25, 2023 09:00 UTC

> “[N]early all network devices had recovered by 09:00 UTC… Final networking equipment recovered by 09:35 UTC.”

Outage over, right?

No… “Due to the WAN impact, our automated systems for maintaining the health of the WAN were paused… some paths in the network experienced increased packet loss from 09:35 UTC until those systems were **manually restarted**...” [emphasis added]

Microsoft Azure Outage

Corrective actions:

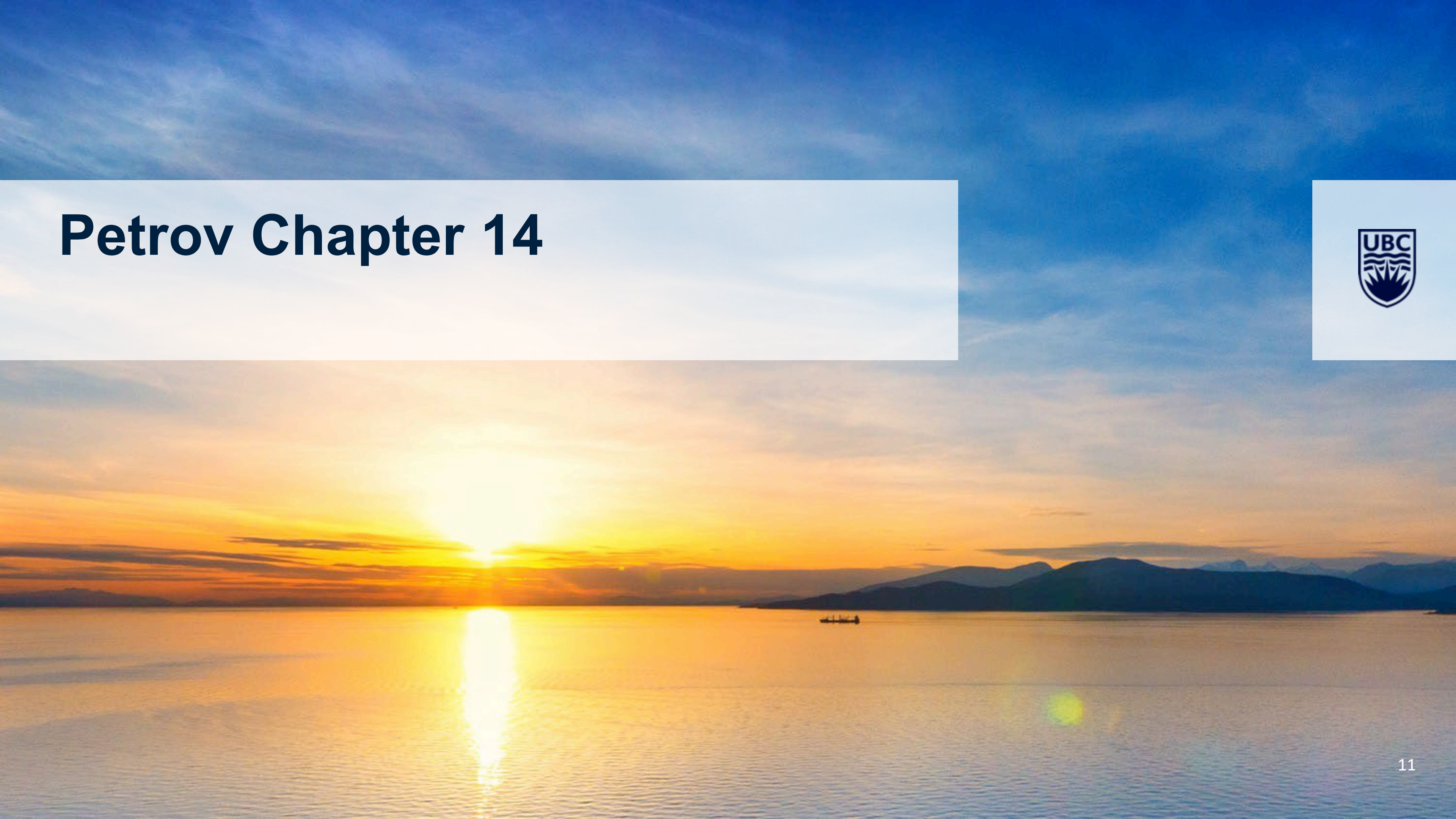
- Blocked highly impactful command from getting executed on the devices
- Require safe change guidelines for command execution

Translation: don’t allow people to do dangerous things.

Source: Azure status history | Microsoft Azure (entry for January 25, 2023)

Notice any patterns for failures?

Petrov Chapter 14



Learning Goals (Petrov Chapter 14)

Learn what a *quorum* is and how it helps with consensus

Understanding why *consensus* is an important tool for building distributed systems

Discuss commonly used consensus algorithms

Understand atomic broadcast

Explore how *consensus* is used in real-world distributed systems.

Quorum

We have a set of *nodes* N

If α and β are subsets of N then:

$\alpha \cap \beta \neq \emptyset$

In other words, they have *at least one common element*.

We call such a set a *quorum*.

Any decision made by one quorum is part of *every* quorum

This is the purpose of consensus: to generate **one** unique decision

Quorum

Further reading:

- Weighted Voting for Replicated Data
- Read-Write Quorum Systems Made Practical

Availability: system can continue to operate with a quorum

Consistency: Intersection property means reads will see most recent writes

- Note: this doesn’t mean all nodes are up to date (“implementation detail”)

Consensus

A mechanism for getting a set of resources controlled by a process to agree on a value.

Distributed systems use consensus to agree on data

- Values in a key/value store
- Entries in a log file

Replicated State Machine (RSM) + Log = “consistent results”

- Here “consistency” just means “gives me the same output in every instance”

Broadcast

Shout out: send a message to *everyone*

No guarantee as to message delivery order

Atomic Broadcast

Reliable Broadcast

Order of delivery is identical

All or none delivery

Examples:

- Virtual Synchrony (Based on ISIS)
- Zookeeper Atomic Broadcast

Paxos

The Part-time Parliament

Proposers: nodes that *propose* a new value

Acceptors: nodes that *vote for* a proposal

Learners: nodes that process accepted proposals

Simple, right?

- Lamport's version

Here's a graphical demonstration (uses PMMC, which is what DSLabs uses)

Paxos (2)

Requirements:

- Only proposed values may be chosen
- Only a single value may be chosen
- Only the chosen value may be learned

Model:

- Nodes operate as fast or slow as they want
- Nodes may fail-stop or fail-restart after a value is chosen
- Messages:
  - Can be slow
  - Can be duplicated
  - Can be lost
  - **Cannot be corrupted**

Paxos (3)

Single proposer can only have one outstanding proposal at a time

- This is done with a sequence number (which is a *string* for “simplicity”)
- Sequence numbers are monotonic
- No node reuses a sequence number for a different proposal

Acceptors:

- Can accept proposals with a higher sequence number
- Ignore proposals with a lower sequence numer

Learners:

- This is the “database” – it *learns* the outcome of the decision
- All learners **must see the same value** (for consistency)

Paxos (4)

Step 1: Proposer sends the proposal to some or all the acceptors (“prepare”)

Step 2: Acceptor receives proposal

- Accepts the proposal “I’ll never accept a lower numbered proposal” (“promise”)
- Identifies the highest numbered proposal it has already accepted (if any)
- **OR** it ignores the proposal (already accepted a higher numbered proposal)

Step 3: Proposer sends an accept message (“I have a quorum”)

Step 4: Acceptor notes the *accepted value*

Step 5: Learner may now *learn* the accepted value (update the database… yeah)

Raft

It’s pretty much Viewstamped Replication

Elect a leader

Leader sends proposals

Acceptors vote

Leader sends accepted messages

This implements a *replicated log* and it is the recovery protocol of this system that is interesting.

Visualizer

Byzantine Fault Tolerance

Allow malicious actors and/or corrupted messages

PBFT: works in the face of up to *f* faulty or lying nodes

- Requires *3f+1* nodes
- Provides efficient mechanism for recovery
- Note: we needed *2f+1* for normal consensus anyway

Use Cases

Distributed Key-Value Store (Anna)

Distributed Message Queue (Kafka)

Distributed File System (Hadoop)

Distributed Caching System (Redis)

Distributed Coordination Service (Apache Zookeeper)

Distributed Graph Processing System (Apache Giraph)

Distributed Stream Processing System (Flink)

Distributed Load Balancer (HAProxy)

Distributed Blockchain Platform (Ethereum)

Questions?

How to use this template

Please note: This template has a variety of slides for your use. To select what slide you would like, click on the drop down menu beside “new slide” button in the top left corner, and pick the corresponding slide. To insert text, simply double click on the text box and start typing. Please be aware that copying and pasting text may change how the font looks. It is better to type directly onto the slide. Also note that larger fonts (size 14+) work better for presentations than smaller sizes. This template uses the font Arial, as PowerPoint users will experience technical difficulties if using UBC’s official fonts. If desired, images can replaced by going into the “Master” view and applying your own image. Please ensure you have the rights to an image before using it.

The following slides are here for visual reference only. Please delete or edit as needed for your own presentation. If you have any questions about how to use this template, please contact UBC Communications and Marketing at comm.marketing@ubc.ca

Insert title here

Insert subtitle here

Name, position

Insert title here

Insert subtitle here

Name, position

Insert title here

Insert subtitle here

Name, position

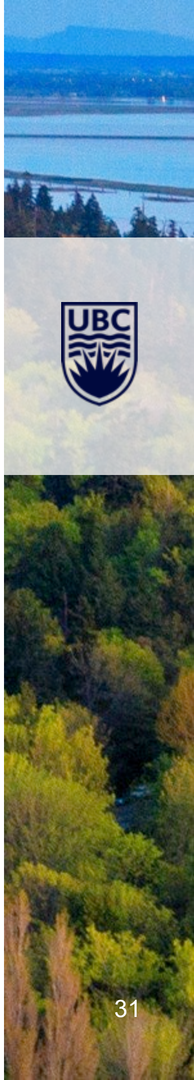
Insert title here

Insert subtitle here

Name, position

Page title

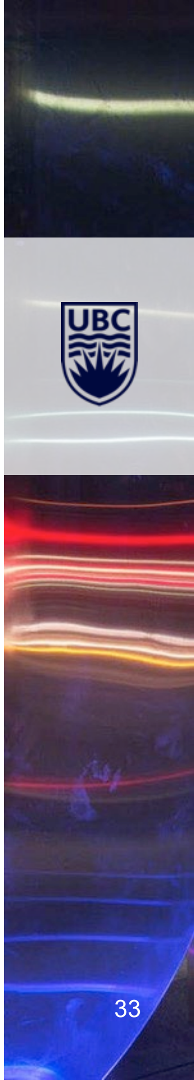
- **Bullet point** list
- **Bullet point** list
- **Bullet point** list
- **Bullet point** list



Insert chapter title

Page title

- **Bullet point** list
- **Bullet point** list
- **Bullet point** list
- **Bullet point** list



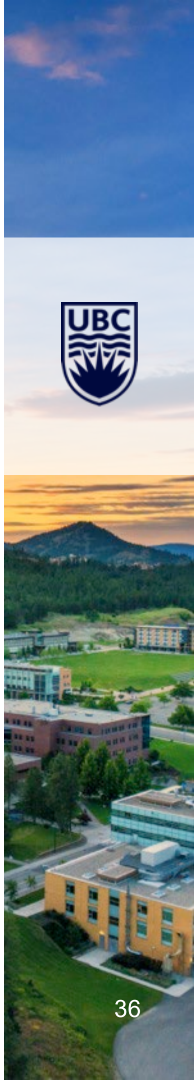
Insert chapter title

Page title

- Bullet point list
- Bullet point list
- Bullet point list
- Bullet point list

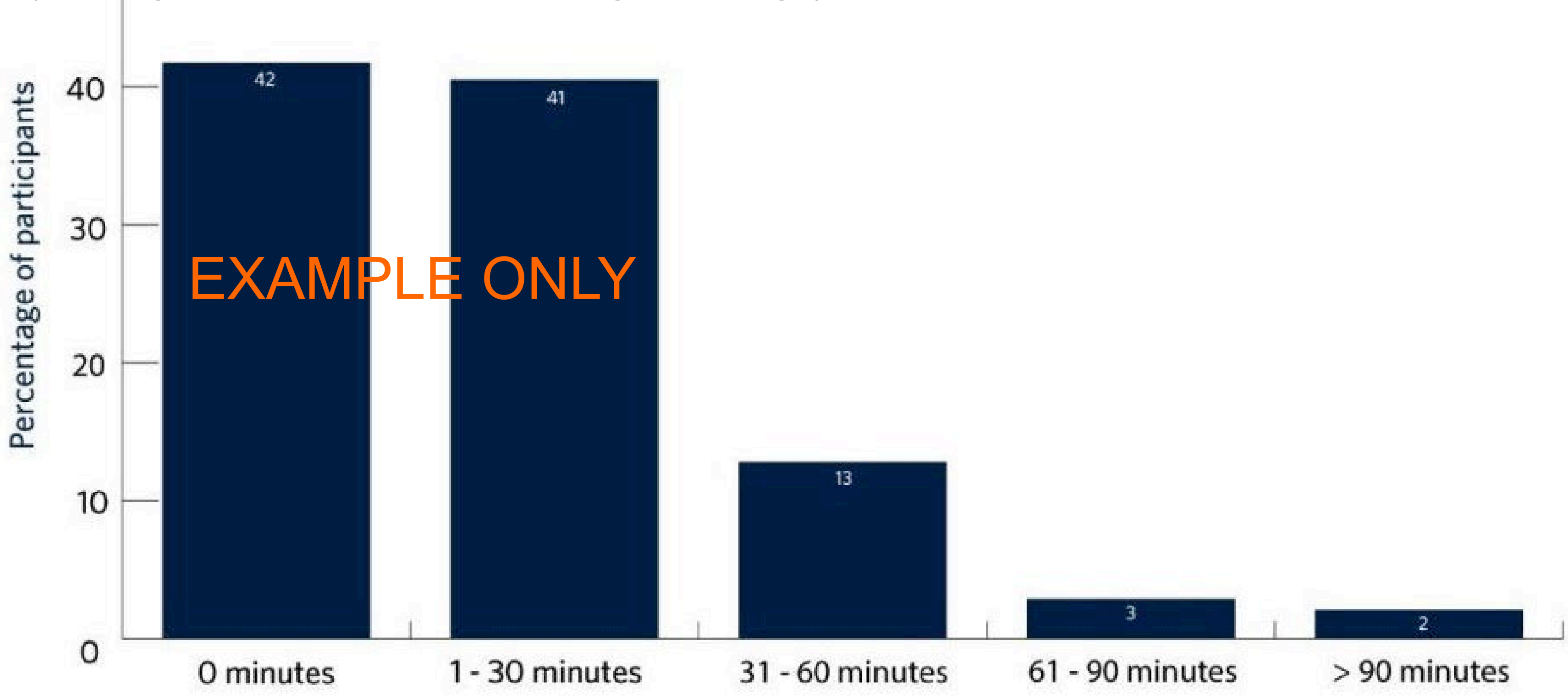
Page title

- **Bullet point** list
- **Bullet point** list
- **Bullet point** list
- **Bullet point** list



Insert title

(delete graph below and insert own graph/image)



EXAMPLE ONLY

THE UNIVERSITY OF BRITISH COLUMBIA

THE UNIVERSITY OF BRITISH COLUMBIA