CSE 486/586 Distributed Systems
Wrap-up

Steve Ko
Computer Sciences and Engineering
University at Buffalo

CSE 486/586, Spring 2013

CSE 486/586, Spring 2013

Building a Distributed System

• “The number of people who know how to build really solid distributed systems...is about ten”
  – Scott Shenker, Professor at UC Berkeley
• Are you confident now?
• What were the most interesting topic to you?

CSE 486/586, Spring 2013

CSE 486/586, Spring 2013

Things We Discussed (Midterm)

• Networking basics (feat. the Internet)
• Failure detection
• Time synchronization
• Logical time & global states
• P2P & DHT
• Reliable multicast
• Consensus basics
• Mutual exclusion & leader election
• RPC

CSE 486/586, Spring 2013

CSE 486/586, Spring 2013

Things We Discussed

• Transactions & concurrency control
• Replication
• Gossiping
• Distributed file systems
• Distributed shared memory
• Paxos
• BFT
• Security

CSE 486/586, Spring 2013

CSE 486/586, Spring 2013

The Way I See It

• We've learned some of the building blocks & fundamental results...
  – Networking basics, failure detection, logical time, reliable multicast, mutual exclusion, leader election, transactions, concurrency control, replication, gossiping, Paxos, BFT, ...
• ...and how real systems get built using those...
  – P2P, DHT, Dynamo, Chubby, ...
• ...and also got some experience in building/using the fundamental building blocks...
  – Ordered multicast for messaging, a DHT, and a replicated key-value storage
Distributed Systems 10 Questions

- Course goal: answering 10 questions on distributed systems
  - At the end of the semester, if you can answer only 10 questions about distributed systems, you'll probably get an A.
  - Easy enough!
- What are those questions?
  - Organized in 6 themes
  - 1~2 questions in each theme
  - A few (or several) lectures to answer each question

Theme 1: Communications

- Q1: how do you talk to another machine?
  - A: Networking basics
  - Know how to use socket now?
- Q2: how do you talk to multiple machines at once?
  - A: Multicast
  - What is “reliable multicast”?
  - What orderings are there for ordered multicast?
- Q3: can you call a function/method/procedure running in another machine?
  - A: RPC
  - What is a stub compiler (generator)?

Theme 2: Concurrency

- Q4: how do you control access to shared resources?
  - A: Distributed mutual exclusion, leader election, etc.
  - Ring election? Modified ring election? Bully algorithm?

Theme 3: Hint

I thought I was doing it...
I'm shaking my tail.
What? I'm doing it too!

I want to shake my tail.
OK
No, I don't want to.
No way!
Theme 3: Consensus

Q5: how do multiple machines reach an agreement?
- A: it’s impossible! (the FLT result), but algorithms do exist that get around the impossibility (Paxos, BFT, etc.)
- What are the phases for Paxos?

Theme 4: Storage Management

Q6: how do you locate where things are and access them?
- A: DHT, distributed file systems, etc.
- Consistent hashing?

Theme 5: Non-Byzantine Failures

Q7: how do you know if a machine has failed?
- A: Failure detection
- What is the fundamental limit of a failure detector?
Q8: how do you program your system to operate continually even under failures?
- A: Replication, gossiping
- Linearizability? Sequential consistency? One-copy serializability?
Theme 6: Byzantine Failures

• Q9: how do you deal with attackers?
  – A: Security
  – What is onion routing?
• Q10: what if some machines malfunction?
  – A: Byzantine fault tolerance
  – To tolerate f faulty nodes, how many nodes do we need in total?

Acknowledgements

• These slides contain material developed and copyrighted by Indranil Gupta (UIUC).