CSE 486/586 Distributed Systems
Byzantine Fault Tolerance

Steve Ko
Computer Sciences and Engineering
University at Buffalo

Recap
• Digital certificates
  – Binds a public key to its owner
  – Establishes a chain of trust
• TLS
  – Provides an application-transparent way of secure communication
  – Uses digital certificates to verify the origin identity
• Authentication
  – Needham-Schroeder & Kerberos

Byzantine Fault Tolerance
• Fault categories
  – Benign: failures we’ve been talking about
  – Byzantine: arbitrary failures
• Benign
  – Fail-stop & crash: process halted
  – Omission: msg loss, send-omission, receive-omission
  – All entities still follow the protocol
• Byzantine
  – A broader category than benign failures
  – Process or channel exhibits arbitrary behavior,
  – May deviate from the protocol
  – Processes can crash, messages can be lost, etc.
  – Can be malicious (attacks, software bugs, etc.)

Result: with \( f \) faulty nodes, we need \( 3f + 1 \) nodes to tolerate their Byzantine behavior.
• Fundamental limitation
• Today’s goal is to understand this limitation.

How about Paxos (that tolerates benign failures)?
• With \( f \) faulty nodes, we need \( 2f + 1 \) (i.e., we need a correct majority.)
• Having \( f \) faulty nodes means that as long as \( f + 1 \) nodes are reachable, Paxos can guarantee an agreement.
• This is the known lower bound for consensus with non-Byzantine failures.

“Byzantine”
• Leslie Lamport (again!) defined the problem & presented the result.
  – “I have long felt that, because it was posed as a cute problem about philosophers seated around a table, Dijkstra’s dining philosopher’s problem received much more attention than it deserves.”
  – “At the time, Albania was a completely closed society, and I felt it unlikely that there would be any Albanians around to object, so the original title of this paper was The Albanian Generals Problem.”
  – “…The obviously more appropriate Byzantine generals then occurred to me.”

Introducing the Byzantine Generals
• Imagine several divisions of the Byzantine army camped outside of a city
• Each division has a general.
• The generals can only communicate by a messenger.
Introducing the Byzantine Generals

- They must decide on a common plan of action.
  - What is this problem?
- But, some of the generals can be traitors.
- Quick example to demonstrate the problem:
  - One commander and two lieutenants
  - With one traitor, can non-traitors decide on a common plan?

Understanding the Problem

- One traitor makes it impossible with three generals.
- Or more generally, when $f$ nodes can behave arbitrarily (Byzantine), $2f + 1$ nodes are not enough to tolerate it.
  - This is unlike Paxos (tolerating non-Byzantine failures).

CSE 486/586 Administrivia

- PA4 due 5/6 (Friday)
- Final: Thursday, 5/12, 8am – 11am at Knox 20
  - Everything
  - No restroom use (this quickly becomes chaotic)

More Practical Setting

- Replicated Web servers
  - Multiple servers running the same state machine.
  - For example, a client asks a question and each server replies with an answer (yes/no).
  - The client determines what the correct answer is based on the replies.
More Practical Setting

• \( f \) Byzantine failures
  – At any point of time, there can be up to \( f \) failures.
• Many possibilities for a failure
  – A crashed process, a message loss, malicious behavior (e.g., a lie), etc., but a client cannot tell which one it is.
  – But in total, the maximum # of failures is bounded by \( f \).

BFT Question

• Given \( f \), how many nodes do we need to tolerate \( f \) Byzantine failures?
  – \( f \) failures can be any mix of malicious servers, crashed servers, message losses, etc.
  – Malicious servers can do anything, e.g., they can lie (if yes, say no, if no, say yes).

Intuition for the Result

• Let’s say we have \( n \) servers, and maximum \( f \) Byzantine failures.
• What is the minimum # of replies that you are always guaranteed to get?
  – \( n - f \)
  – Why? \( f \) maximum failures can all be crashed processes

Intuition for the Result

• The problem is that a client does not know what kinds those \( f \) failures are.
• Upon receiving \( n - f \) replies (guaranteed), can the client tell if the rest of the replies will come?
  – No, \( f \) faults might all be crashed processes. But what does this mean?

Intuition for the Result

• This means that if a client receives \( n - f \) replies, the client needs to determine what the correct answer is at that time. The rest of the replies might never come.
• Upon receiving \( n - f \) replies, how many replies can come from malicious servers (i.e., lies)?
  – Still \( f \), since some servers can just be really slow.

Intuition for the Result

• What can be the minimum \( n \) to determine the correct answer? What if \( n = 2f + 1 \)?
  – It doesn’t work.
• How can we make it work?
  – If we make sure that \( n - f \) replies always contain more replies from honest nodes than Byzantine nodes, we’re safe.
Intuition for the Result

• How can we make sure that \( n - f \) replies always contain more replies from honest nodes than Byzantine nodes?
  – We set \( n = 3f + 1 \)
  – We can always obtain \( n - f \) votes, i.e., \( 2f + 1 \) votes. Then we have at least \( f + 1 \) votes from honest nodes, one more than the number of potential faulty nodes.

Write/Read Example

• One client writes to X.
• A malicious node omits it.
• Another client reads X.
• It can still get the latest write.

Summary

• Byzantine generals problem
  – They must decide on a common plan of action.
  – But, some of the generals can be traitors.

• Requirements
  – All loyal generals decide upon the same plan of action (e.g., attack or retreat).
  – A small number of traitors cannot cause the loyal generals to adopt a bad plan.

• Impossibility result
  – In general, with less than \( 3f + 1 \) nodes, we cannot tolerate \( f \) faulty nodes.

Acknowledgements

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