The Byzantine Generals Problem

by Leslie Lamport, Robert Shostak, and Marshall Pease

- Fault tolerant computing
- Diagnosis
  - testing
  - location
  - repair
- Reliability
  - will it work in the presence of faults?

Analogy...

- Byzantine generals
  - each general observes the enemy and communicates his observations to the others
  - "attack","retreat"
- Loyal
- Traitors
- Loyal to reach agreement
  - with **oral** messages: solvable if 2/3 are loyal
  - with unforgeable **written** messages: the problem is solvable for any number of generals and traitors

- Two conditions

  A. All loyal generals decide upon the same plan of action

  B. A small number of traitors cannot cause the loyal generals to adopt a bad plan.
- Let $v(i)$ be the information communicated by the i^{th} general
- There are n generals
- Each general combines $v(1), ..., v(n)$ into a single plan of action
- For condition **A** to be satisfied the following must be true:
  - every loyal general must obtain the same information $v(1), ..., v(n)$
  - careful: traitor may send different values to different generals

  - If the i^{th} general is loyal, then the value that he sends must be used by every loyal general as the value of $v(i)$
  - alternatively, any two loyal generals use the same value of $v(i)$
- How the *single* general communicates his value to all others
- "Commanding general"

IC1. All loyal lieutenants obey the same order

IC2. If the commanding general is loyal, then every loyal lieutenant obeys the order he sends

Interactive consistency conditions

- Back to the original problem...

Solution with Oral Messages

- Assumptions

A1. Every message that is sent is delivered correctly

A2. The receiver of a message knows who sent it

A3. The absence of a message can be detected

Algorithm:

- Commander sends his value to everyone else
- If no message, timeout and use a default (RETREAT)
- Each lieutenant, in turn, sends his value to everybody else (except commander)
- Each of these in turn, sends to everyone else (except commander and first lieutenant)
- ... ETC...

- At each level/calling, each general applies the majority function and reports results to the caller
- Number of messages:
  - commander sends out $N-1$ messages
  - each lieutenant sends out $N-2$ messages which trigger $N-3$ messages, etc.
  - m levels, where m is the maximum number of traitors
- Total: $(N-1)(N-2)...(N-m)$ = O(N^m)

Solution with Signed Messages

- Allow generals to send unforgeable signed messages

A4: Assumption:

(a) A loyal general's signature cannot be forged and any alteration of the contents of his signed messages can be detected

(b) Anyone can verify the authenticity of a general's signature

- No assumption about a traitor's signature – we even allow traitors to cooperate
- The commander sends a signed order to each of his lieutenants
- Each lieutenant then adds his signature to that order and sends to the other lieutenants who add theirs and send it to others, etc

- $v : j : i \rightarrow$ value v signed by j and then value $v : j$ signed by i
- Each lieutenant maintains a set V_i of properly signed orders
- Algorithm works like OM, each lieutenant stops at the appropriate place (after a stream has been received or time-out)
- If the commander is loyal, then this set should not contain more than one element (could have many messages, but a single order)
- The lieutenants will know if a commander is a traitor because his signature will appear at two different orders