COMP28112 – Lecture 15
Byzantine fault tolerance: dealing with arbitrary failures
The Byzantine Generals' problem (Byzantine Agreement)

Overview

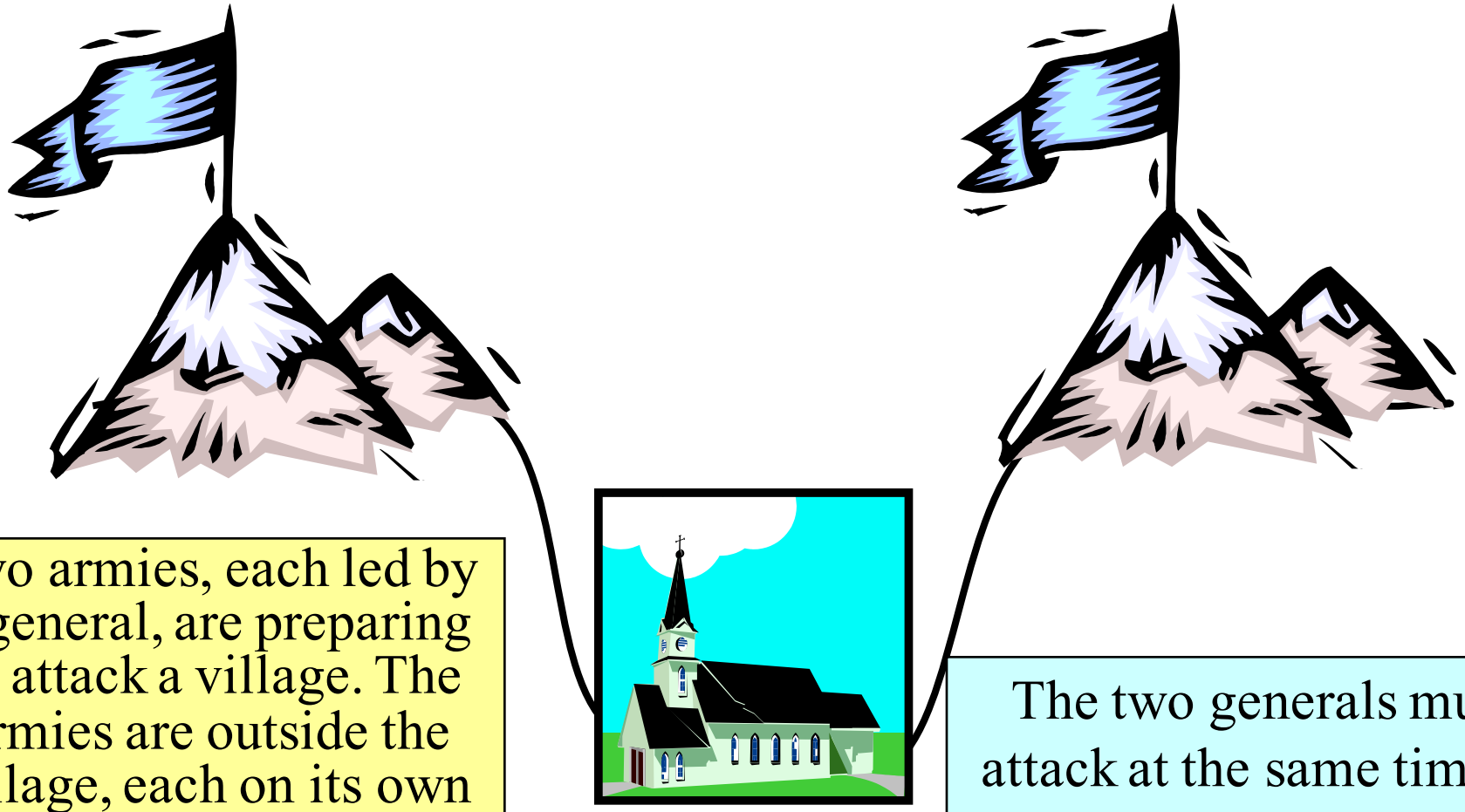
Can unanimity be achieved in an unreliable distributed system?

- Dealing with communication uncertainty
  - Messages are lost due to the unreliability of the communication channels
- Dealing with faulty components
  - Some components of the distributed system might give wrong answers

Still, how can we reach an agreement?

The two generals' problem (or paradox)…

(pitfalls and challenges of communication with unreliable links…)



Two armies, each led by a general, are preparing to attack a village. The armies are outside the village, each on its own hill. The generals can communicate only by sending messengers passing through the valley

The two generals must attack at the same time to succeed!

http://en.wikipedia.org/wiki/Two_Generals'_Problem

The two generals' problem

- Two armies, surrounding a city, are prepared to attack the city. They must attack at the same time in order to succeed (otherwise they will fail).
  - Simplest form: one general decides the time and communicates this to the other general. Both generals should be able to conclude: "we will both attack at the same time" – Impossible!
- They send messages to each other through an unreliable medium. E.g.:
  - General1 to General2: Let us attack at 9am
  - General2 to General1: I received your plan for attack

  General2 doesn't know if General1 received the message.

  Bottom line: there is no way to guarantee that both generals agree a message was delivered!

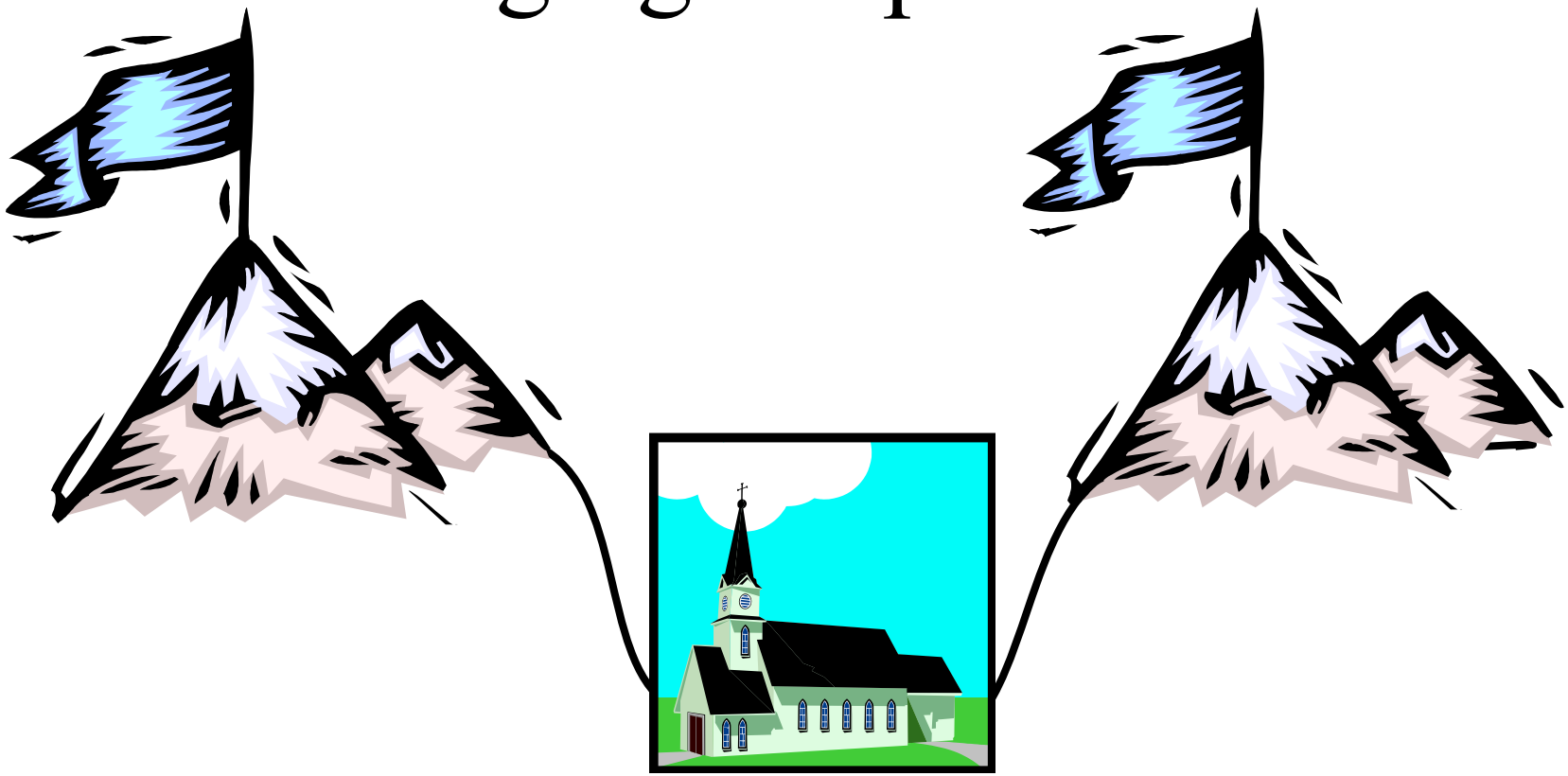
Sketch of a proof of the impossibility result

- Suppose there is any sequence of messages, some successfully delivered, some not.
- Take a subset of successfully delivered messages (with at least one message indicating the time/plan)
- Consider the last message in this sequence
- If the last message had not been received, the receiver would have decided not to attack. However, from the viewpoint of the sender, nothing has changed!
- We have built a scenario where one general will attack but the other will not!

Engineering Solutions

- We need to accept the uncertainty of the communications channel! We cannot eliminate it!
- If the leading general sends 1000 messages with the time to attack, the probability of all the messages being lost is low (in general, p^m, where p is the probability of a single message being lost and m is the number of messages)
- If the receiving general sends a 1000 messages acknowledging receipt of a plan, then, again, the probability of all the messages being lost is low!
- There is no algorithm that will guarantee that one general will attack without the other. However, from practical purposes, if the chances of this happening is, say, 0.00001, we are close be certain!
- NB: If the communications channel was reliable, only one message would suffice!

Changing the problem…



Assume a **reliable** communications channel.

What if one general is a traitor? (and knowingly transmits wrong messages?)

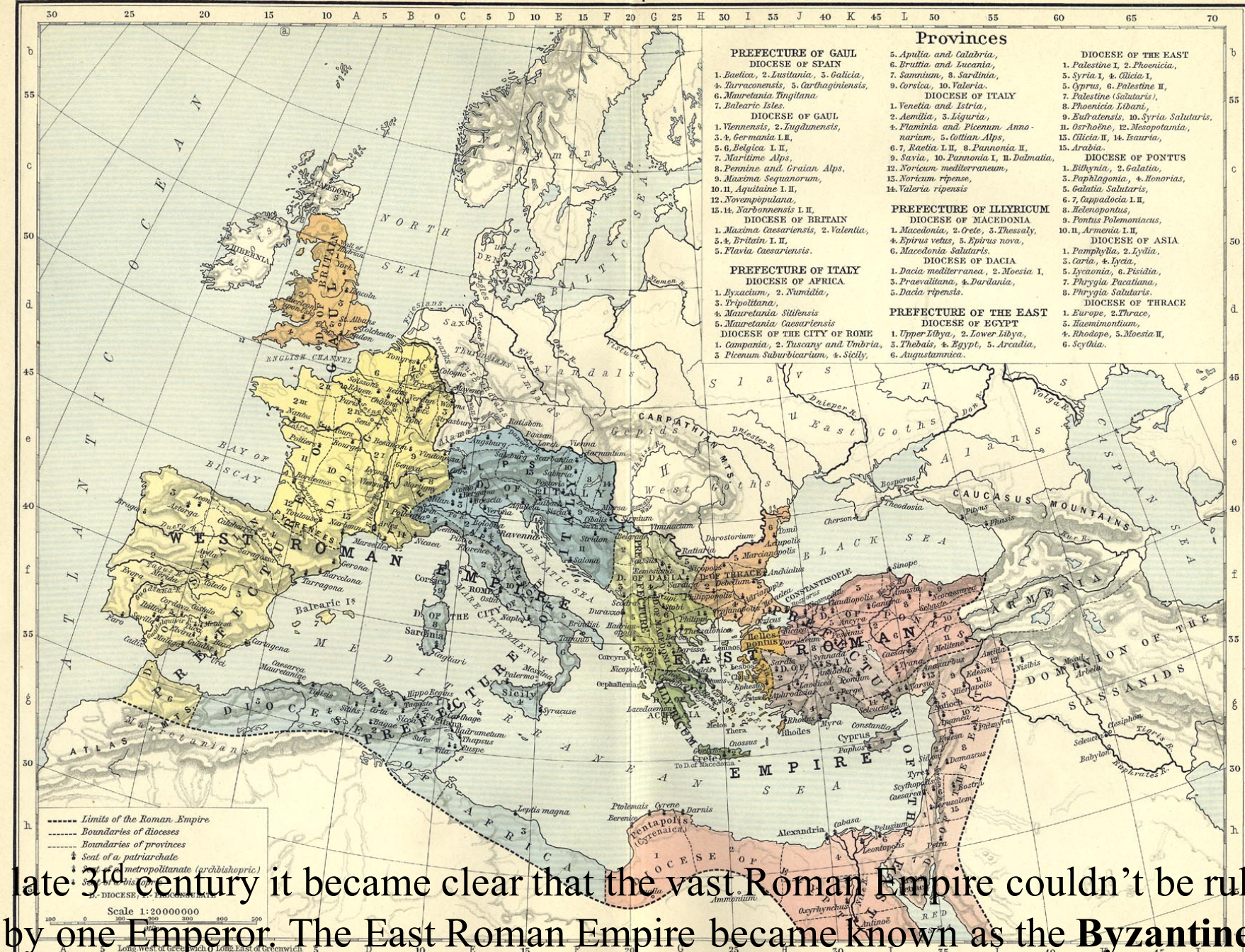
Would it be possible to reach an agreement with 3 generals? (even if one of them was a traitor)

Byzantine failures

- Components not only behave erroneously, but also fail to behave consistently when interacting with other components.
- It's not only about message loss any more! It is about getting inconsistent answers!

The worst possible type of failure!

- Dealing with such problems relates to the sub-field of **byzantine fault tolerance**.



In late 3rd century it became clear that the vast Roman Empire couldn't be ruled by one Emperor. The East Roman Empire became known as the **Byzantine**

On ‘byzantinism’

- A term with rather negative connotations, suggesting intrigues and plots, coined in the 19th century and reflecting biases at the time.
- *“Ever since our rough crusading forefathers first saw Constantinople and met, to their contemptuous disgust, a society where everyone read and wrote, ate food with forks and preferred diplomacy to war, it has been fashionable to pass the Byzantines by with scorn and to use their name as synonymous with decadence”* (Steven Runciman)
- http://en.wikipedia.org/wiki/Byzantinism

Byzantine Generals' Problem

(Lamport et al., ACM TOPLAS, July 1982)
http://research.microsoft.com/users/lamport/pubs/byz.pdf

Several divisions of the byzantine army are camped outside an enemy city, each division commanded by its own general. The generals can communicate with one another only by messenger… they must decide upon a common plan of action (e.g., attack or retreat). *However, some of the generals may be traitors, trying to prevent the loyal generals from reaching an agreement.*

The generals must have an algorithm to guarantee that:

- All loyal generals decide upon the same plan of action
- A small number of traitors cannot cause the loyal generals to adopt a bad plan.

General solution and reformulation

- Each general sends a message to all other generals.
- All generals combine their information and majority ignores minority.
- However, if the loyal generals are to decide upon the same plan of action, they should receive the same messages from other generals. Thus, the problem can be reduced to how a single general sends his value to the others.
- **Byzantine Generals Problem**: A commanding general must send an order to the *n-1* lieutenant generals such that:
  - All loyal lieutenants obey the same order
  - If the commanding general is loyal, then every loyal lieutenant obeys the order of the commanding general.

Impossibility results

- 3 generals, one traitor among them
- 2 types of messages: attack or retreat
- Agreement cannot be reached

Case 1

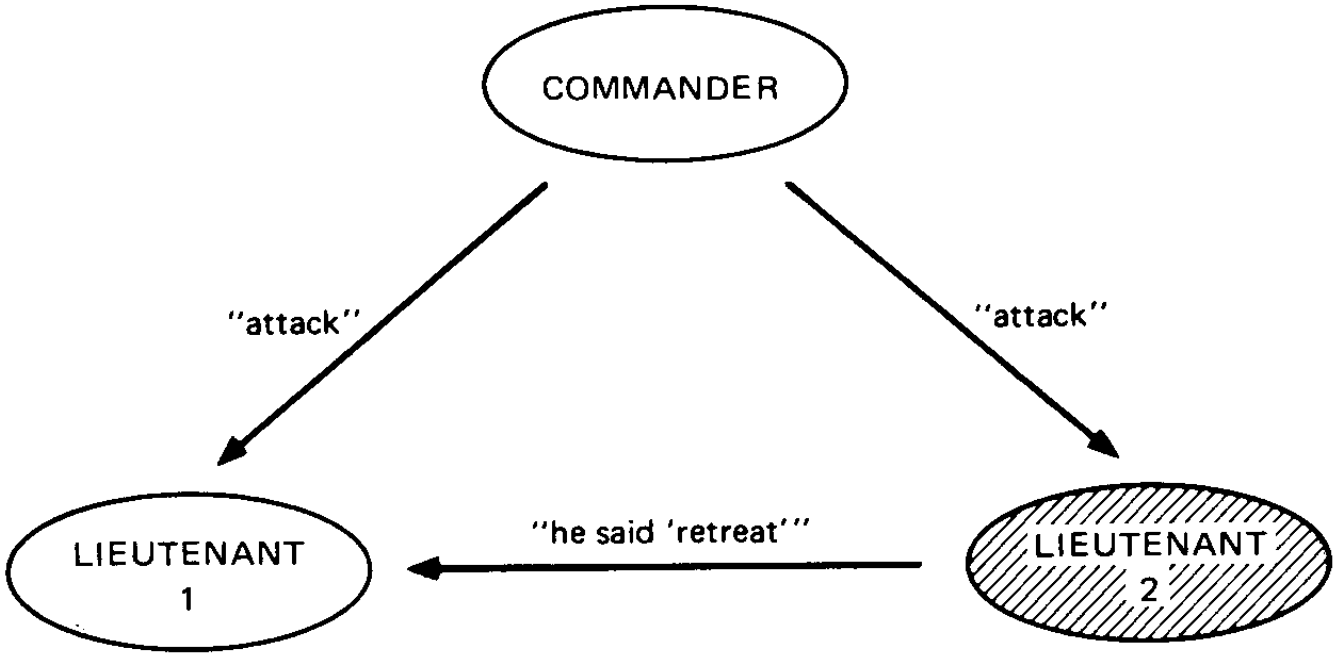


Fig. 1. Lieutenant 2 a traitor.

- Lieutenant 1 knows that one of the other two is a traitor, but he must obey the order of the commanding general.

Case 2

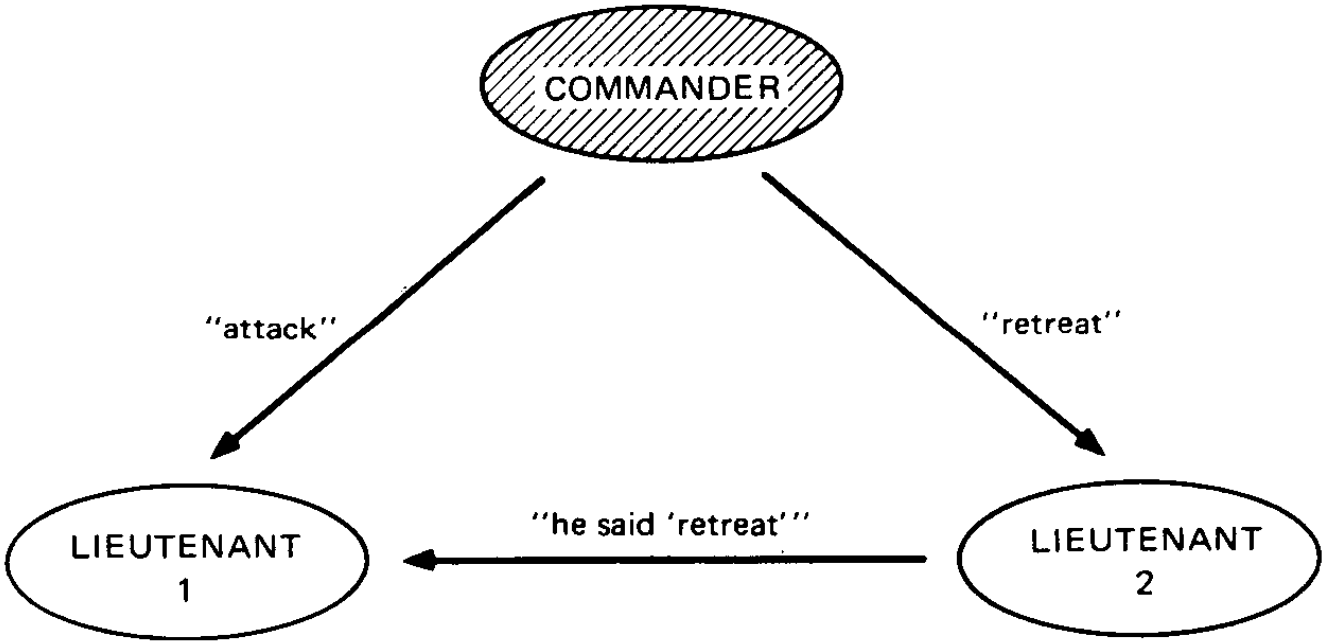


Fig. 2. The commander a traitor.

- Lieutenant 1 does not know who the traitor is. Each lieutenant must obey the order.

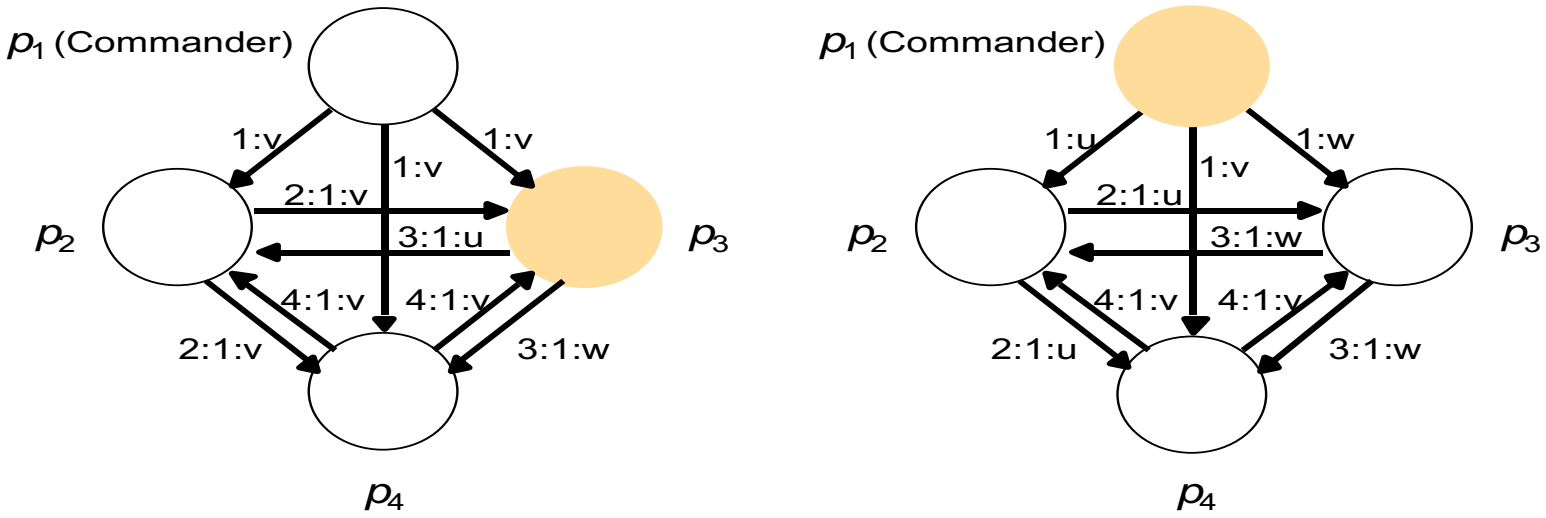
Remarks

- The evidence we have from these two cases does not constitute a full proof!
- It is not easy to describe the full proof (i.e., that no agreement can be reached with 3 generals in the presence of a traitor), but there is one. (see Journal of ACM, Apr.1980)
- In general, it can be proved that for n components (generals), no solution is possible if $n \leq 3f$, where f is the number of faulty components (traitors).
- If $n > 3f$, then see next…

A (sketch of an) algorithm for agreement

- Commander sends an order to each lieutenant
- Each lieutenant sends this order to all other lieutenants
- Decision is made on the basis of majority

Figure 12.20
Four byzantine generals



Faulty processes are shown coloured

The full story

(for information only…see Lamport *et al*'s 1982 paper for more)

- m is the number of traitors
- n is the number of generals
- The algorithm operates in $m+1$ rounds and involves sending $O(n^{m+1})$ messages

---

Algorithm OM(0).

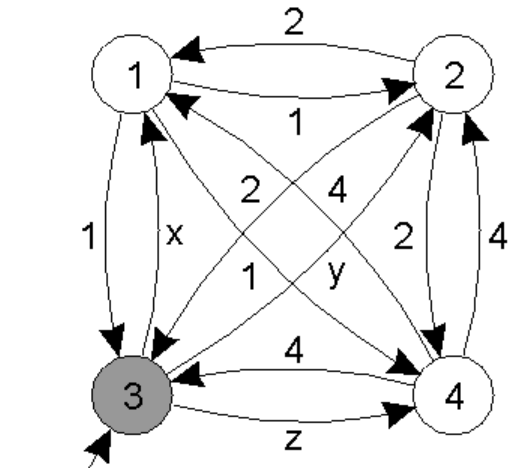
(1) The commander sends his value to every lieutenant.
(2) Each lieutenant uses the value he receives from the commander, or uses the value RETREAT if he receives no value.

Algorithm OM(m), $m > 0$.

(1) The commander sends his value to every lieutenant.
(2) For each i, let v_i be the value Lieutenant i receives from the commander, or else be RETREAT if he receives no value. Lieutenant i acts as the commander in Algorithm OM($m - 1$) to send the value v_i to each of the $n - 2$ other lieutenants.
(3) For each i, and each $j \neq i$, let v_j be the value Lieutenant i received from Lieutenant j in step (2) (using Algorithm OM($m - 1$)), or else RETREAT if he received no such value. Lieutenant i uses the value *majority*$(v_1, \ldots, v_{n-1})$.

Agreement in Faulty Systems (1)

(see Tanenbaum, figure 8-5)



(a)

1 Got(1, 2, x, 4)
2 Got(1, 2, y, 4)
3 Got(1, 2, 3, 4)
4 Got(1, 2, z, 4)

(b)

1 Got	2 Got	4 Got
(1, 2, y, 4)	(1, 2, x, 4)	(1, 2, x, 4)
(a, b, c, d)	(e, f, g, h)	(1, 2, y, 4)
(1, 2, z, 4)	(1, 2, z, 4)	(i, j, k, l)

(c)

- The Byzantine generals problem for 3 loyal generals and 1 traitor.

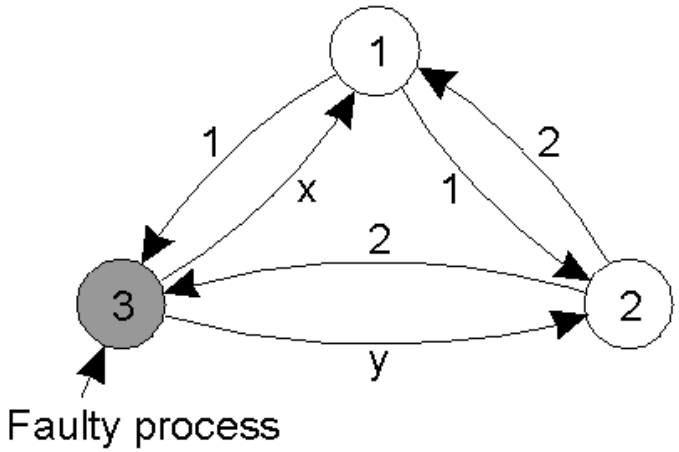
a) The generals announce their troop strengths (in units of 1 kilosoldiers).

b) The vectors that each general assembles based on (a); this is propagated to all other generals.

c) The vectors that each general receives in step 3; there is an agreement about 1,2,4.

Agreement in Faulty Systems (2)

(see Tanenbaum Figure 8-6)



(a)

1	Got(1, 2, x)
2	Got(1, 2, y)
3	Got(1, 2, 3)

(b)

1 Got	2 Got
(1, 2, y)	(1, 2, x)
(a, b, c)	(d, e, f)

(c)

- The same as in previous slide, except now with 2 loyal generals and one traitor.
- No agreement is possible!

Other issues…

- One can use signed messages to make sure that messages from loyal generals cannot be forged.
- What if not all generals can reach all other generals directly?
  - (they may need to go through traitors…)
- What if messages are not delivered?
  - Failure both in communication and components

Literature

- There is a vast amount of work in the distributed systems literature related to byzantine problems
- For example, scholar.google.com returns:
  - More than 25K for each of the search terms *byzantine generals* (4860 if searching using "…"), *byzantine fault tolerance* (4530), *byzantine failures* (4480) and *byzantine agreement* (5160)

    (ok, some of these references, might be really about byzantine history and civilisation, not computer science! ☺)

Conclusions

- Solutions to byzantine (arbitrary) failures may be expensive… One way to think about the solution(s) is that they use **redundancy** and **majority voting** to achieve reliability…
- Are *3m+1* replicas for *m* failures too many?
  - There are tradeoffs between the need for reliability and performance.
- Determining *m* in a real system might not be easy…
- However: arbitrary failures in real systems may be rare or can be isolated…
- **Reading**: CDK4, page 501, and Section 12.5.3 (pages 504-507); CDK5, Section 15.5.3; Tanenbaum *et al*, pp. 332-335.