The Byzantine Generals Problem

Leslie Lamport, Robert Shostak & Marshall Pease

Presentation Structure

1. Introduction
2. Problem Definition
3. Impossibility
4. Solutions
    a. Oral Messages
    b. Signed Messages
5. Missing Communication paths
6. Reliable Systems
7. Conclusion

1. Introduction

- How can you achieve Byzantine Agreement in a system in case of a failure?
- What is the Byzantine Generals Problem?
- What are some solutions of the Byzantine Generals Problem?

2. Problem Definition

Problem

- A computer system with multiple components
- A failure occurs in one of the components
- The failed component may send misleading messages to other components (Byzantine!)

Goal

Build a reliable computer system that copes with failures of its components

What is the Byzantine Generals Problem?

An abstract representation of the problem described

Establishing the Byzantine Generals Problem

- Enemy city

- Divisions of Byzantine Army camped outside, each led by a Lieutenant Commander

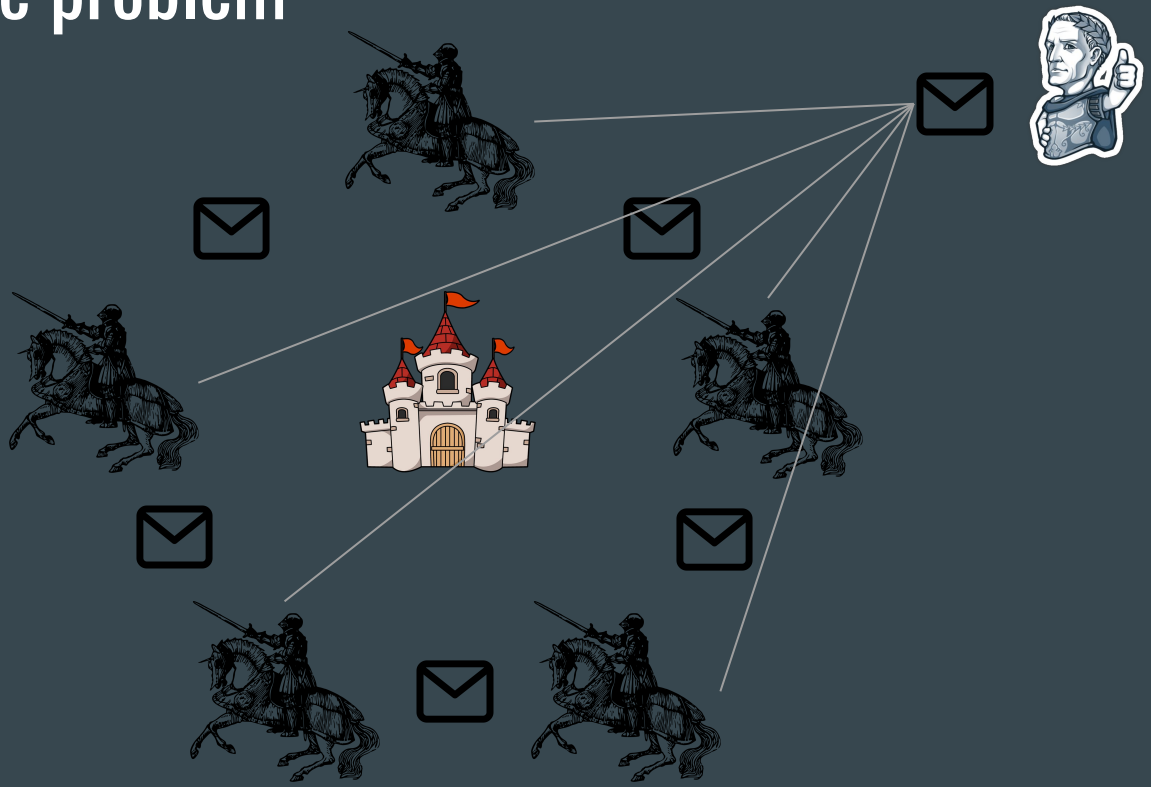
- A General Commander giving orders to Lieutenant Commanders

- The Generals can communicate with one another only through a messenger

The Generals need to decide upon a common plan of action!

Establishing the problem

Establishing the problem

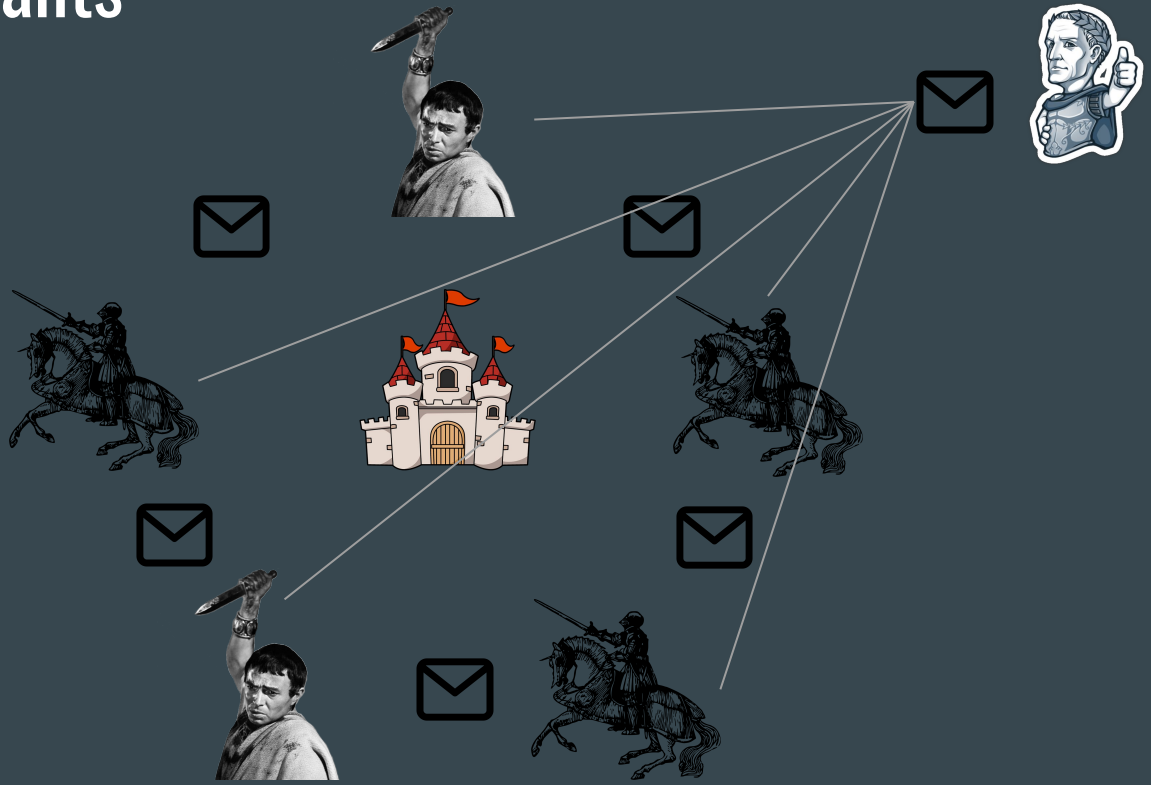


Establishing the problem

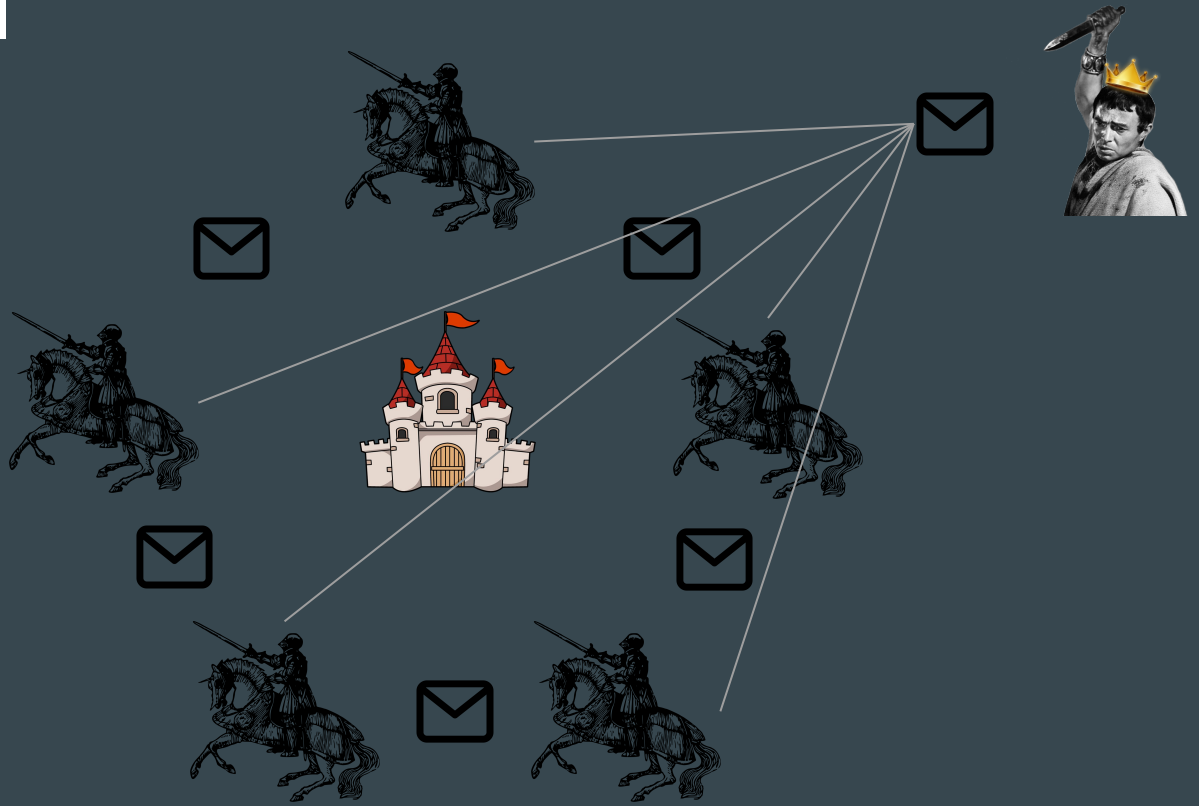
What happens in the case of:

Brutus AKA **traitors**?

Traitor Lieutenants



Traitor General



Goal Conditions

An algorithm is needed to guarantee that:

A. All loyal Generals decide upon the same plan of action
   - The traitors can do anything they wish
   - The loyal Generals should reach agreement and agree upon a reasonable plan

B. A small number of traitors cannot cause the loyal Generals to adopt a bad plan
   - ‘Bad plan’ is hard to formalize
   - Consider how the Generals reach a decision

Problem Definition Recap

- Each General observes the enemy & communicates his observations to the others.
- Let **u(i)** be the information communicated by the **i**-th General.
- Each General uses some methods for combining the values **u(1), ... , u(n)** into a single plan of action.
- Traitors may send different values to different Generals.
- We do not want a general to use a value **u(i)** different from the one sent by the **i**-th General when the **i**-th General is loyal.

Problem Definition

In order for conditions A and B to stand:

1. Any two loyal Generals use the same value of **u(i)**
2. If the **i**-th General is loyal then the value that he sends must be used by every loyal General as the value of **u(i)**

Problem Definition

Conditions 1 & 2 are both conditions on the single value sent by the **i**-th General

Our consideration should focus on how a single General sends his values to the others

This is phrased as a Commanding General sending an order to his lieutenants

Finally - Byzantine Generals Problem Definition

A Commanding General must send an order to his **n-1** Lieutenant Generals such that:

- All loyal Lieutenants obey the same order. (**IC1**)
- If the Commanding General is loyal, then every loyal Lieutenant obeys the order he sends. (**IC2**)

IC1 & **IC2** are called the interactive consistency conditions.

3. Impossibility

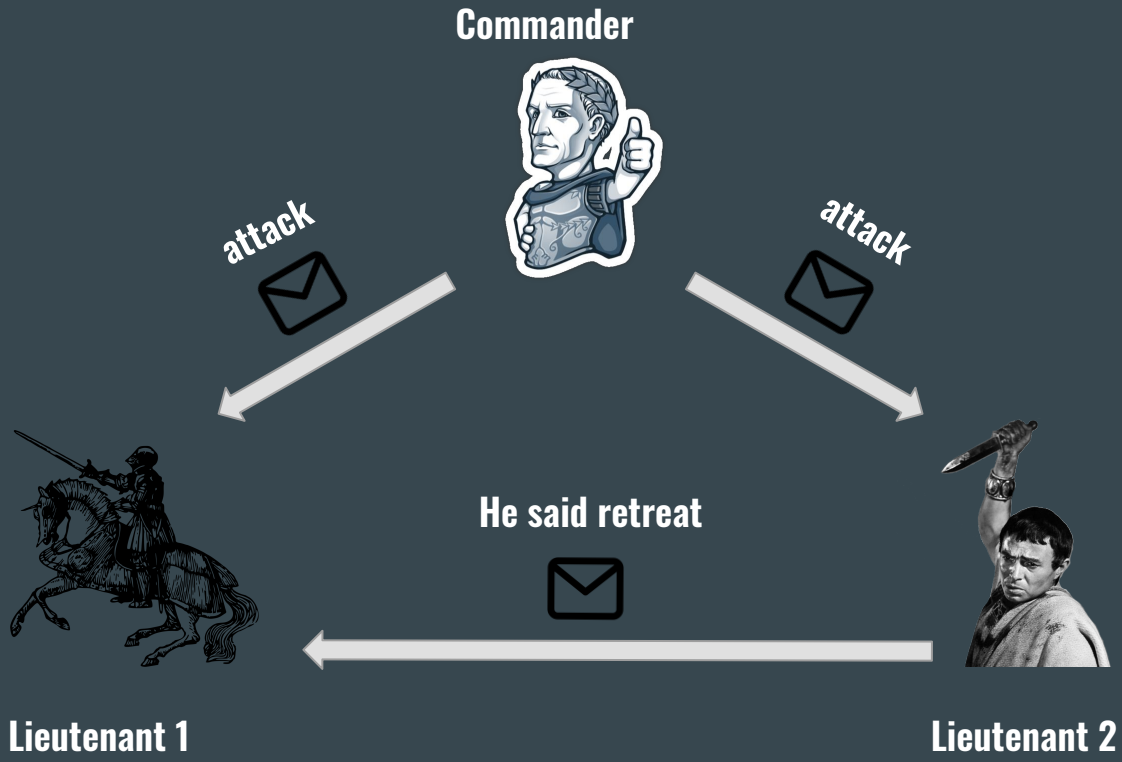
Impossibility

- In the problem discussed Generals contact through oral messages.
- The contents of the oral messages are under the control of the sender who can forge them

With only three Generals no solution can work in the presence of a single traitor

Scenarios with 3 Generals

V = {attack, retreat}

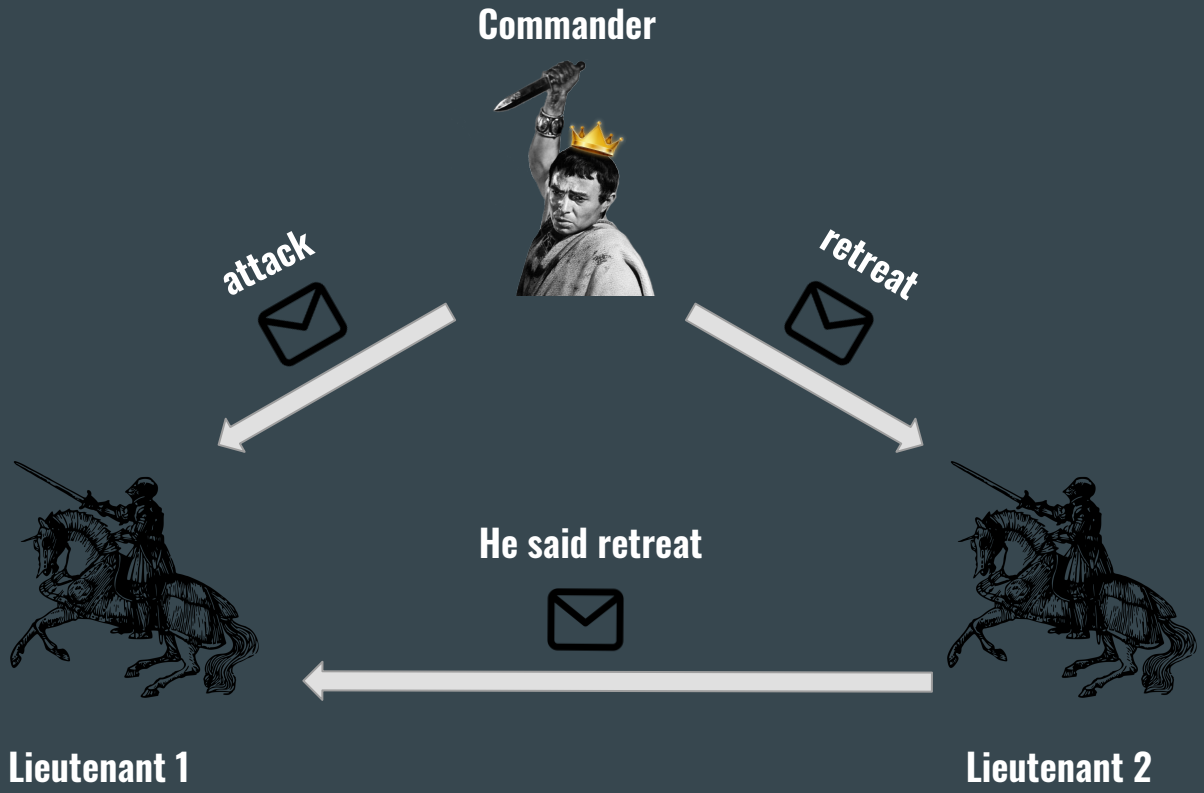


Scenarios with 3 Generals

Because of **IC2**, Lieutenant 1 must obey the order to attack **(a)**

Scenarios with 3 Generals

V = {attack, retreat}



Scenarios with 3 Generals

- Lieutenant 1 obeys the **attack** order.
- Lieutenant 2 obeys the **retreat** order.
- **IC1** is violated **(b)**

Because of **(a), (b)** NO solution works for three generals in the presence of one traitor!

Scenarios with 3 Generals

- There is no way for Lieutenant 1 to distinguish between these two situations
- Lieutenant 1 must obey the Commander and attack in both scenarios

3m + 1 Generals

- Using that we can show that no solution with fewer than **3m + 1** Generals can cope with **m** traitors.
- Proof by contradiction in the next slides.

Proof by contradiction

- Suppose that exists a solution for fewer than 3m + 1 Generals where m are traitors.
- We are going to use this solution to construct a three General solution to the Byzantine Generals Problem that works with one traitor, which as we know is impossible.
- The Generals of the assumed solution are called **Albanian Generals**.
- The Generals of the constructed solution are called **Byzantine Generals**.

Proof by contradiction

1. Each Byzantine General is simulating at most **m** Albanian Generals
   a. The Byzantine Commander simulates the Albanian Commander & at most **m-1** Lieutenants
   b. The two other Byzantine Generals simulate each **m** Albanian Generals
   c. Only one Byzantine General can be the traitor which means at most **m** Albanian Generals
2. IC1 holds hence all the Albanian lieutenants being simulated by a loyal Byzantine Lieutenant obey the same order.
3. IC2 holds hence if the Albanian Commander is loyal then every Albanian Lieutenant obeys the order he sends.
4. Since IC1 and IC2 hold for Albanian Generals they will apply to the Byzantine ones
5. We proved that a solution for 3 Byzantine Generals exist!

No solution with fewer than 3m + 1 Generals can cope with m traitors

Altering the problem

- Instead of a plan of attack, Generals must agree only upon an approximate time of attack!
- Our Byzantine General Problem Changes to the following.

Approximate Byzantine Generals Problem

A Commanding General must send an order denoting an approximate time of attack to his **n-1** Lieutenant Generals such that:

- All loyal Lieutenants attack within 10 minutes of one another. (**IC1'**)
- If the Commanding General is loyal, then every loyal Lieutenant attacks within 10 minutes of the time given in the Commander's order. (**IC2'**)

IC1' & **IC2'** are called the interactive consistency conditions.

Impossibility

- Like our original problem this problem is unsolvable unless more than the two thirds of the Generals are loyal.
- Proof by contradiction

Commander General

- Orders an attack by sending an attack time of 1:00
- Orders a retreat by sending an attack time of 2:00

Lieutenant General

1. After receiving the attack order from the Commander:
   a. If the time <= 1:10 **attack**
   b. If the time >= 1:50 **retreat**
   c. Otherwise **2**
2. Ask other Lieutenant what decision they took in **1**
   a. If the other Lieutenant reached a decision, make the same decision he did
   b. Otherwise **retreat**

Proof by contradiction

- Assume there is a solution for this problem with three Generals coping with one traitor.
- Using this assumption, construct a three-General coping with one traitor solution to the Byzantine Generals Problem which we know is impossible.

Proof by contradiction

- From **IC2'**: If the Commander is loyal then a loyal Lieutenant will obtain the correct order in step 1. Hence, **IC2** is satisfied.
- If the Commander is loyal then **IC1** follows from **IC2**.
- If the Commander isn't loyal this means that both Lieutenants are loyal, so they both reach the same decision. Hence **IC1** is satisfied.
- We proved that Byzantine Generals Problem with three Generals coping with one traitor has a solution.

Reaching approximate agreement is just as hard as reaching exact agreement

4a. A Solution With Oral Messages

The Byzantine Generals Problem using oral messages can be solved if there are at least 3m + 1 Generals coping with m traitors

Definition of Oral Messages

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.

A1, A2 : Prevent a traitor from interfering with the communication between two Generals

A3: Will foil a traitor who tries to prevent a decision by simply not sending messages.

Default behavior is **RETREAT**.

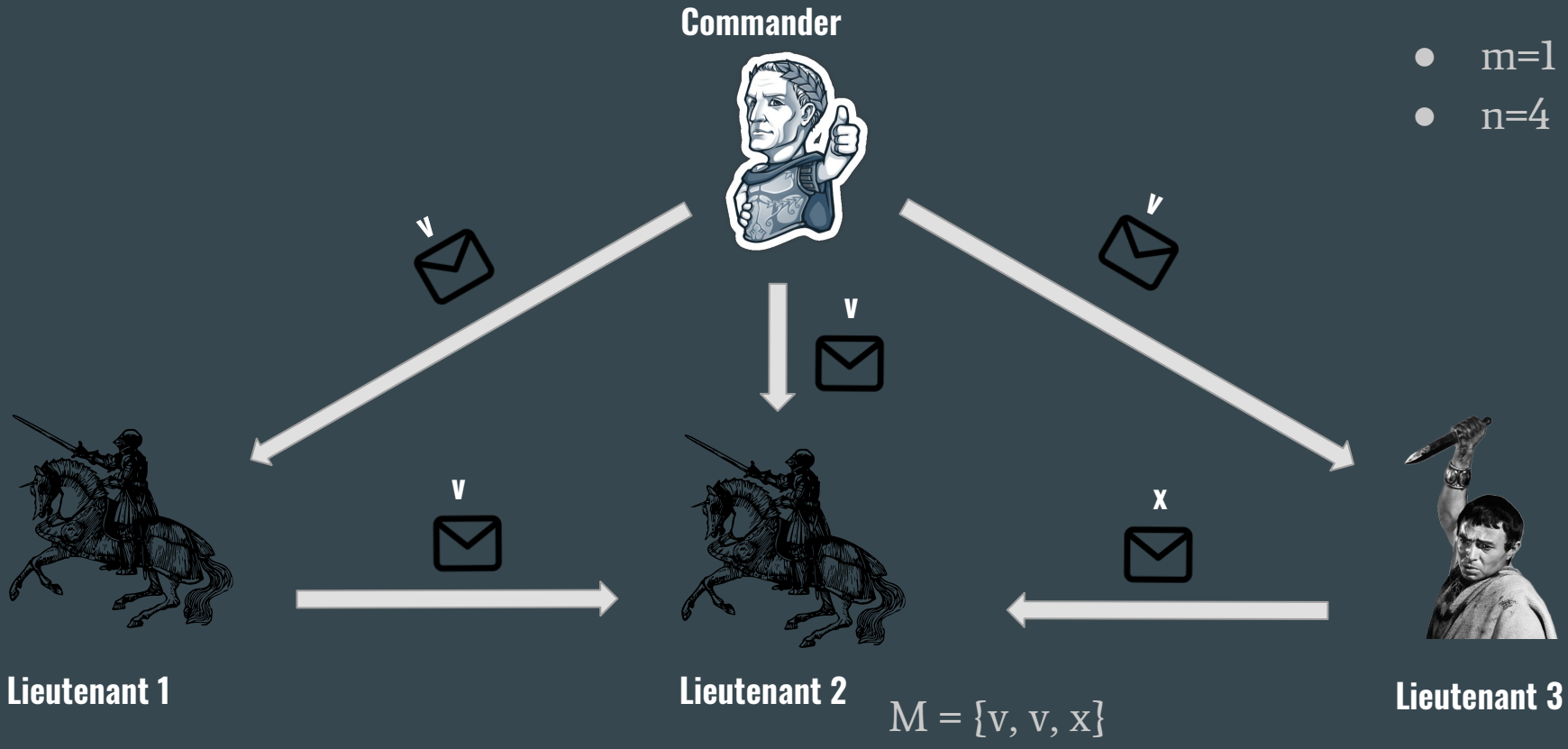
Oral Message Algorithm - OM(m)

- Algorithm OM(0)
  1. The Commander General sends his value to every Lieutenant General.
  2. Each Lieutenant General uses the value he receives from the Commander General.
  3. If no such order is received he uses the value **Retreat.**

Oral Message Algorithm - OM(m)

- Algorithm OM(m), m>0
  1. The Commander General sends his value to every Lieutenant General.
  2. For each i, let u_i be the value Lieutenant General i received from the Commander General or else Retreat if he receives no value.
  3. Lieutenant General i acts as the Commander General in **OM(m-1)** to send the value u_i to each of the n-2 other Lieutenant Generals.
  4. Lieutenant performs a majority voting on the votes he obtained.

Oral Message Algorithm - OM(m) Example

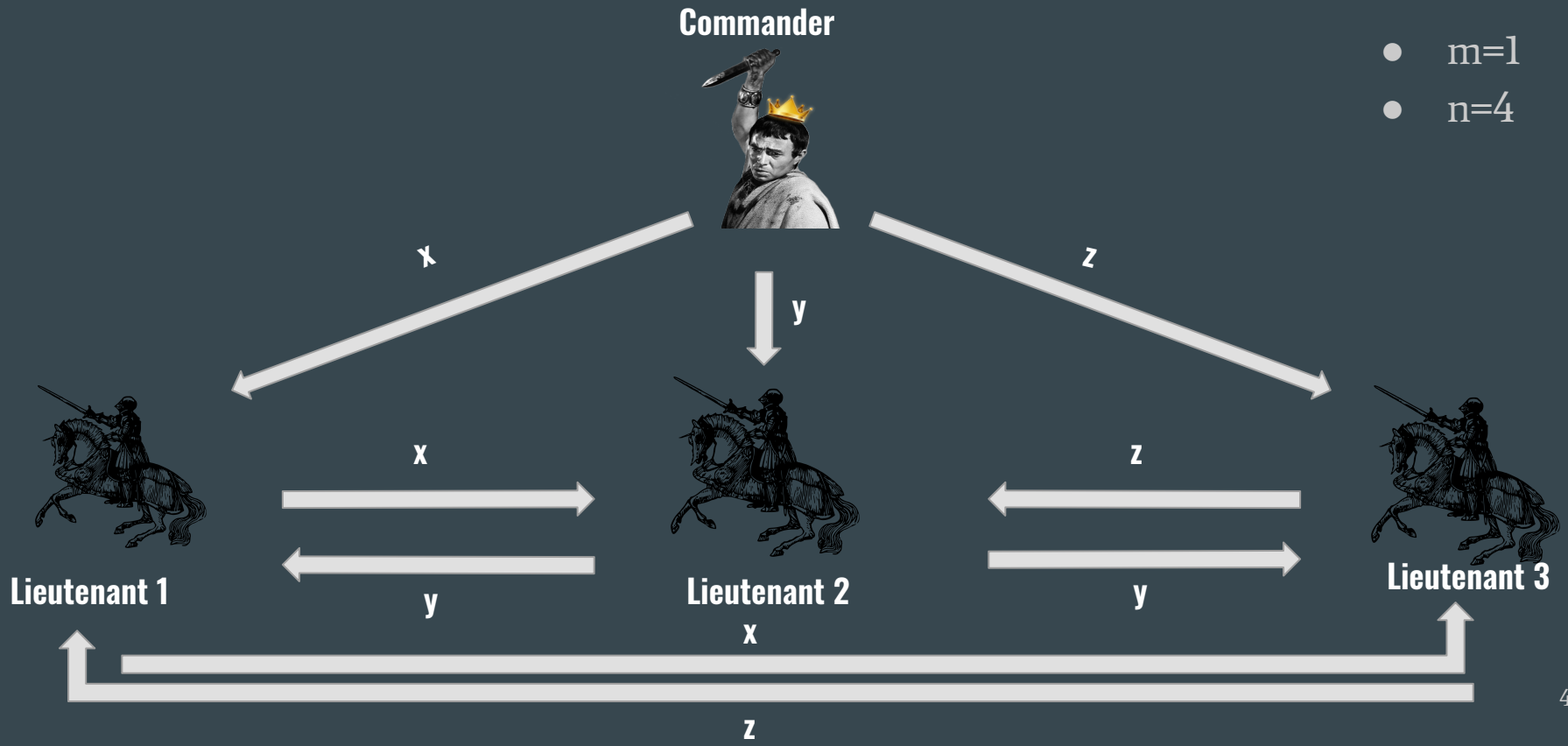


- m=1
- n=4

M = {v, v, x}

Oral Message Algorithm - OM(m) Example

- m=1
- n=4



For any m, Algorithm OM(m) satisfies conditions IC1 and IC2 if there are more than 3m Generals and at most m traitors.

4b. A Solution With Signed Messages

A Solution With Signed Messages

- The difficulty of the problem lies in the fact that the traitor Generals can ‘lie’.
- ‘Lie’ means to send false messages.
- The problem becomes much easier if this ability is restricted.

Definition of Oral Messages

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.

A1, A2 : Prevent a traitor from interfering with the communication between two Generals

A3: Will foil a traitor who tries to prevent a decision by simply not sending messages.

Default behavior is **RETREAT.**

Definition of Unforgeable Signed Messages

We add to A1 - A3:

A4.

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.

A three-General solution coping with one traitor now exists!

Problem Definition

- The Commander General issues a signed order to each of his Lieutenants.
- Each Lieutenant makes multiple copies of the order, signs it and sends it to the other Lieutenants.
- There is no majority voting for the order. Each Lieutenant obeys an order based on a **choice** function, which is applied to a set of orders to obtain a single one.
- The value **x** signed by General **i** is denoted as **x : i**. Thus, **u : j : i** denotes the value **u** signed by **j** and then that value **u : j** signed by **i**.

Signed Message Algorithm - SM(m)

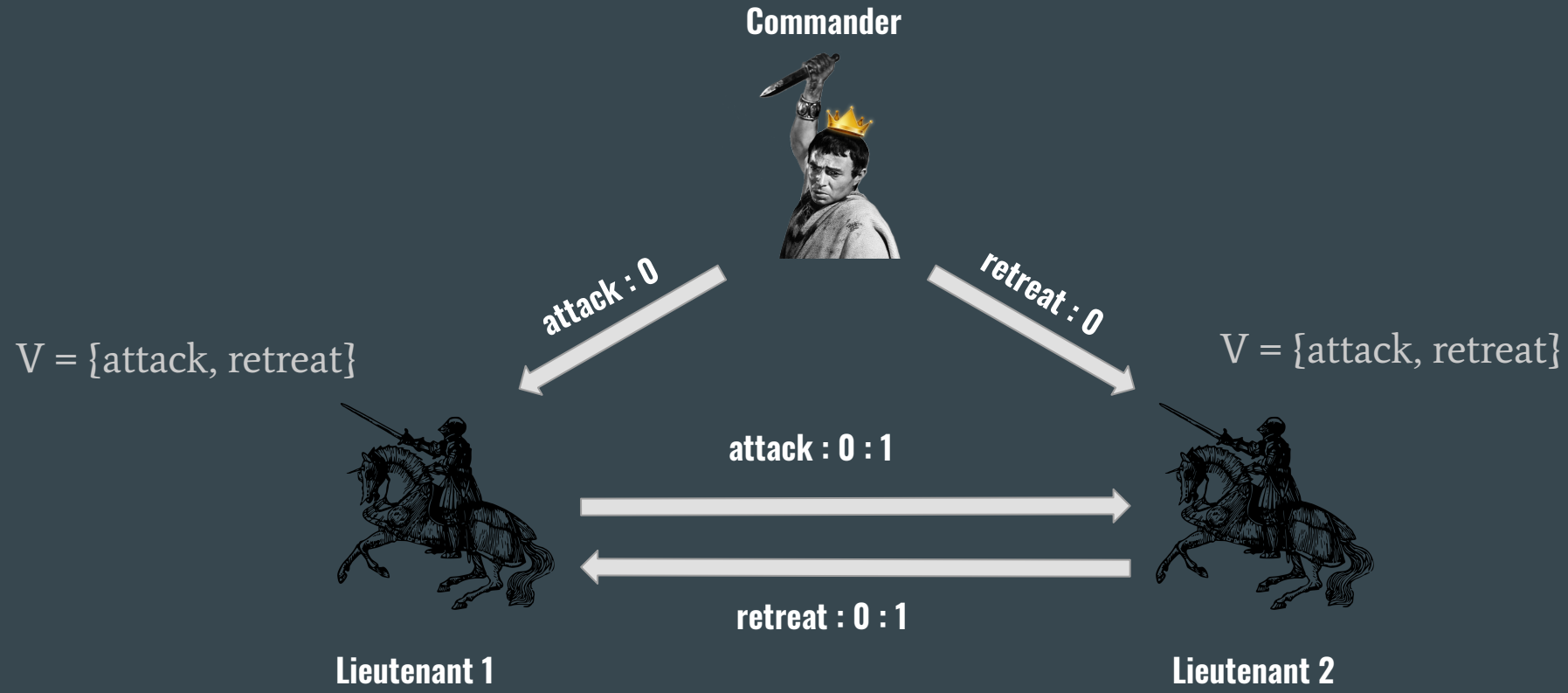
Initially, Vi = ∅

1. The commander signs and sends his value to every lieutenant.
2. For each **i**:
   A. If Lieutenant **i** receives a message of the form **u : 0** from the commander & he has not yet received any order, then
      i. He lets Vi equal {u};
      ii. He sends the message **u : 0 : i** to every other Lieutenant.
   B. If Lieutenant **i** receives a message of the form **u : 0** : j_i : ... : j_k & **u** is not in the set **Vi**, then
      i. He adds **u** to **Vi**
      ii. If **k < m** he sends the message **u : 0** : j_i : ... : j_k to every other Lieutenant other than j_i : ... : j_k

Signed Message Algorithm - SM(m)

3. For each **i:** When Lieutenant **i** will receive no more messages, he obeys the order **choice(Vi)**.

Signed Message Algorithm - SM(m) Example



For any m, Algorithm SM(m) solves the Byzantine Generals Problem if there at most m traitors

5. Missing Communication Paths

Missing Communication Paths

- All this time we assume that a General can send messages directly to every other General.
- We now remove this assumption.
- Support there are physical barrier that restrict the communication between Generals.

P-Regular Graphs

A. A set of nodes $\{i_1, \dots, i_p\}$ is said to be a regular set of neighbours of a node **i** if:
   a. Each $\mathbf{i_j}$ is a neighbour of i
   b. For any General **k** different from **i**, there exists paths $\gamma_{j,k}$ from $\mathbf{i_j}$ to **k** not passing through **i** such that any two different paths $\gamma_{i,k}$ have no node in common other than **k**.

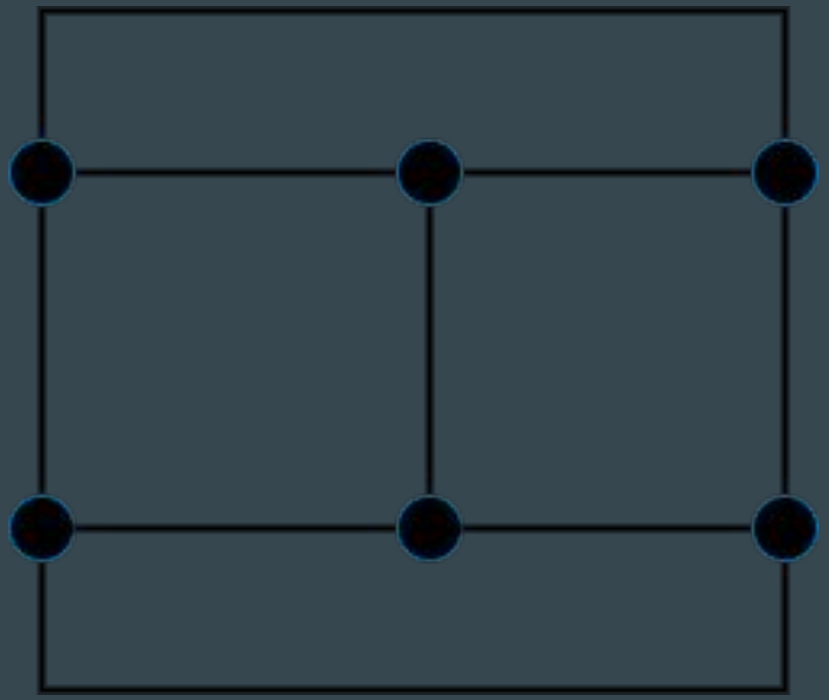
B. The Graph G(V, E) is said to be **p-regular** if every node has a regular set of neighbours consisting of **p** distinct nodes.

P-Regular Graphs

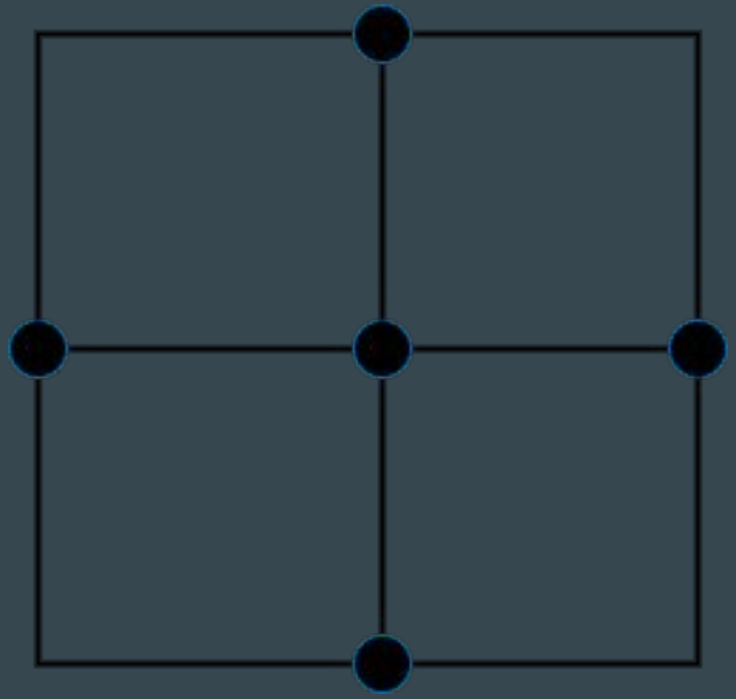
To put in plain words:

A graph G is said to be p-**regular** if every node has **p** distinct neighbours.

P-Regular Graph Example



A Not P-Regular Graph



Extending OM(m)

Extending the algorithm described earlier to solve the Byzantine Generals Problem in the presence of **m** traitors if the graph G of Generals is **3m-regular**.

Oral Message Algorithm for p-regular G - OM(m, p)

1. Choose a regular set N of neighbours of the Commander consisting of p Lieutenants.
2. The Commander sends his value to every Lieutenant in N.
3. Let $\mathbf{u_i}$ be the value Lieutenant **i** receives from the Commander. Lieutenant **i** sends $\mathbf{u_i}$ to every other Lieutenant **k** as follows:
   a. If **m = 1**, then sends the value along the path $\gamma_{i,k}$ which is guaranteed to exist based on the Definition given before.
   b. If **m > 1**, then acts as Commander using the Algorithm **OM(m - 1, p - 1)**.
4. Lieutenant **k** makes a choice based on the majority vote of the orders obtained or chooses RETREAT if he received no value.

For any $m > 0$ and any $p \geq 3m$, Algorithm OM(m, p) solves the Byzantine Generals Problem if there are at most m traitors.

Extending SM(m)

- Extension for OM(m) required the graph G to be 3m-regular.
- This is a strong connectivity hypothesis.
- If there are 3m + 1 Generals this means complete connectivity
  OM(m, 3m) == OM(m)
- SM(M) is extended to allow the weakest possible connectivity hypothesis.

The weakest connectivity hypothesis for which the Byzantine Generals Problem is solvable is that the subgraph formed by the loyal generals be connected.

Signed Message Algorithm for p-regular G - SM(m)

Initially, Vi = ∅

1. The commander signs and sends his order to the neighbouring Lieutenants.
2. For each **i**:
   A. If Lieutenant **i** receives a message of the form **u : 0** from the commander & he has not yet received any order, then
      i. He lets Vi equal {u};
      ii. He sends the message **u : 0 : i** to every other Lieutenant.
   B. If Lieutenant **i** receives a message of the form **u : 0** : $\mathbf{j_i}$ **: ... :** $\mathbf{j_k}$ & **u** is not in the set **Vi**, then
      i. He adds **u** to **Vi**
      ii. If **k < m** he sends the message **u : 0** : $\mathbf{j_i}$ **: ... :** $\mathbf{j_k}$ to every neighbouring Lieutenant other than $\mathbf{j_i}$ **: ... :** $\mathbf{j_k}$

Signed Message Algorithm for p-regular G - SM(m)

3. For each **i:** When Lieutenant **i** will receive no more messages, he obeys the order **choice(Vi)**.

6. Reliable Systems

Achieving Reliability

- The implementation of a reliable computer system involves using several different processors to compute the same result and then perform a **majority vote** on their output to obtain a single value.
- Majority voting is reliable only if the nonfaulty processors produce the same output.
- The input data that the components obtain for processing may be faulty due to malfunctioning or synchronizing.
    - The master component may give different values to different processors.
    - Different processes can get different values from a nonfaulty input if they read the value while it is changing.

Achieving Reliability

In order for the majority voting to yield a reliable system the following conditions need to be satisfied:

1. All nonfaulty processors must use the same input value.
2. If the input unit is nonfaulty, then all nonfaulty processes use the value it provides as input.

These are the interactive consistency conditions **IC1** & **IC2** from the problem we tackled before.

Achieving Reliability

- The processors communicate among themselves, in order to guarantee that they will get the same value from an input device.
- In case of a faulty value by the master component the processors acquire a reasonable input value.

Applying Solutions Given to Computer Systems

In order for the solutions to apply to computer systems, there must be a messaging passing system that meets assumptions A1-A3 or A1-A4 for algorithm SM(m).

Definition of Oral - Signed Messages

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.
A4.
    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.

Assumption A1

A1. Every message that is sent is delivered correctly.

- In real systems communication lines can fail.
- For OM Algorithms the failure of a processor or a communication line is the same thing. These algorithms will work in the presence of up to **m** failures.
- The SM Algorithms are insensitive to communication failure, since the graph G lowers its connectivity.

Assumption A2

A2. The receiver of a message knows who sent it.

- A faulty processor should not be able to impersonate a non faulty one.
- The IPC should be over fixed lines.

Assumption A3

A3. The absence of a message can be detected.

- The absence of a message in a computer system can only be detected if it fails to arrive within some time limit.
- There needs to be a fixed maximum time limit for the generation and transmission of a message.
- The sender and receiver need to have clocks to synchronize within some fixed maximum error.

Assumption A4

A4.

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.

- Property **(a)** can never be guaranteed since faulty processor could generate any data item *Si*.
- Based on the faults we expect to counter (1. Random Malfunction 2. Malicious Intelligence) we can make the probability of the violation of property **(a)** as small as we wish.
- For **(1)** when a processor malfunctions we can reduce the probability of it making a valid signature.
- For **(2)** the construction of the signature is a cryptographic problem.

7. Conclusion

Conclusion

- Discussed what are the key errors to reach Byzantine Agreement in a system.
- Defined an abstract problem that describes them.
- Discussed several solutions for the respective abstract problem.
- Using the results of the problem discussed we set constraints for a reliable computer system.

Thank you!

Any questions?