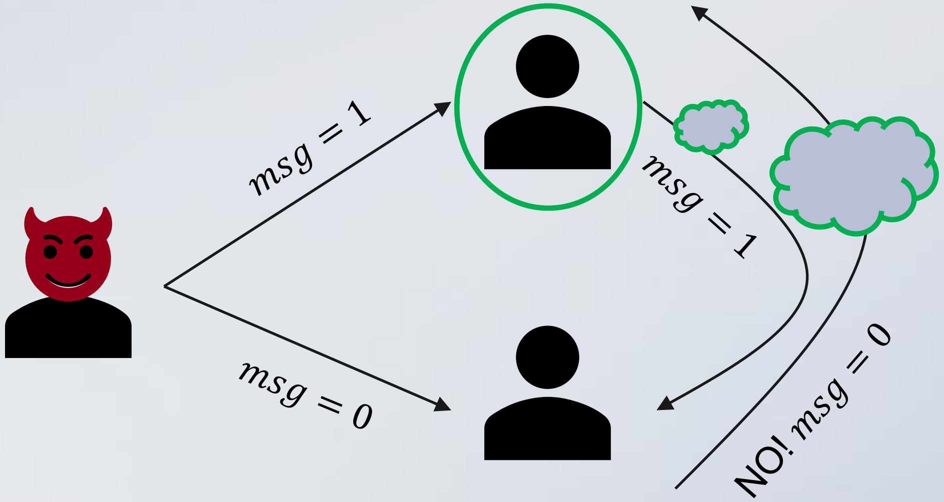
Byzantine Reliable Broadcast

An Adaptive Protocol for the Fault Detection in the Authenticated Double-Echo Broadcast

Giuseppe Daidone 2122594

Byzantine Process

- **Take control over an algorithm**
- Act as they want
- May compromise the behavior of correct processes
- **Difficult to detect**, assumptions on number of Byzantine needed



Who is the impostor?

Byzantine in Broadcast Communication

- **Authenticated Double-Echo Broadcast**
  - Byzantine Reliable Broadcast implementation
  - Known process set Π
  - Known number of processes N
  - Known number of Byzantine f
- Main Phases: Send, Echo, Ready, Deliver
- What if f is unknown?

Designing a New Protocol

- How to determine f?
- The **goal** is to **design a new protocol**, based on the Authenticated Double-Echo Broadcast, that **allows processes to detect Byzantine and estimate f**
- The new protocol will be called **Adaptive**

Assumptions

- **Every process** knows the **number of processes** into the system N
- At the beginning, **every process is considered correct** $correct = \Pi, f = 0$
- Cryptographic communication, through **Authenticated Perfect Point-to-Point Links**
- **Initial trust** of the sender s (until ready phase)
- Byzantine processes **can act as they want**

An Adaptive Solution(1)

- Every process **analyse its echo set and ready set**, trying to find **heterogeneity**
- Then, set a **consistent message** as the **message more frequent** in the echo set, excluding the sender (**may happen symmetry** and process would be stuck)
- Example:
  - $\Pi = \{P_0, P_1, P_2, P_3\}, s = P_0$
  - $P_3 : \{P_0 : m, P_1 : m, \boxed{P_2 : m'}, P_3 : m\}_{Echo}$
  - $cons = m$

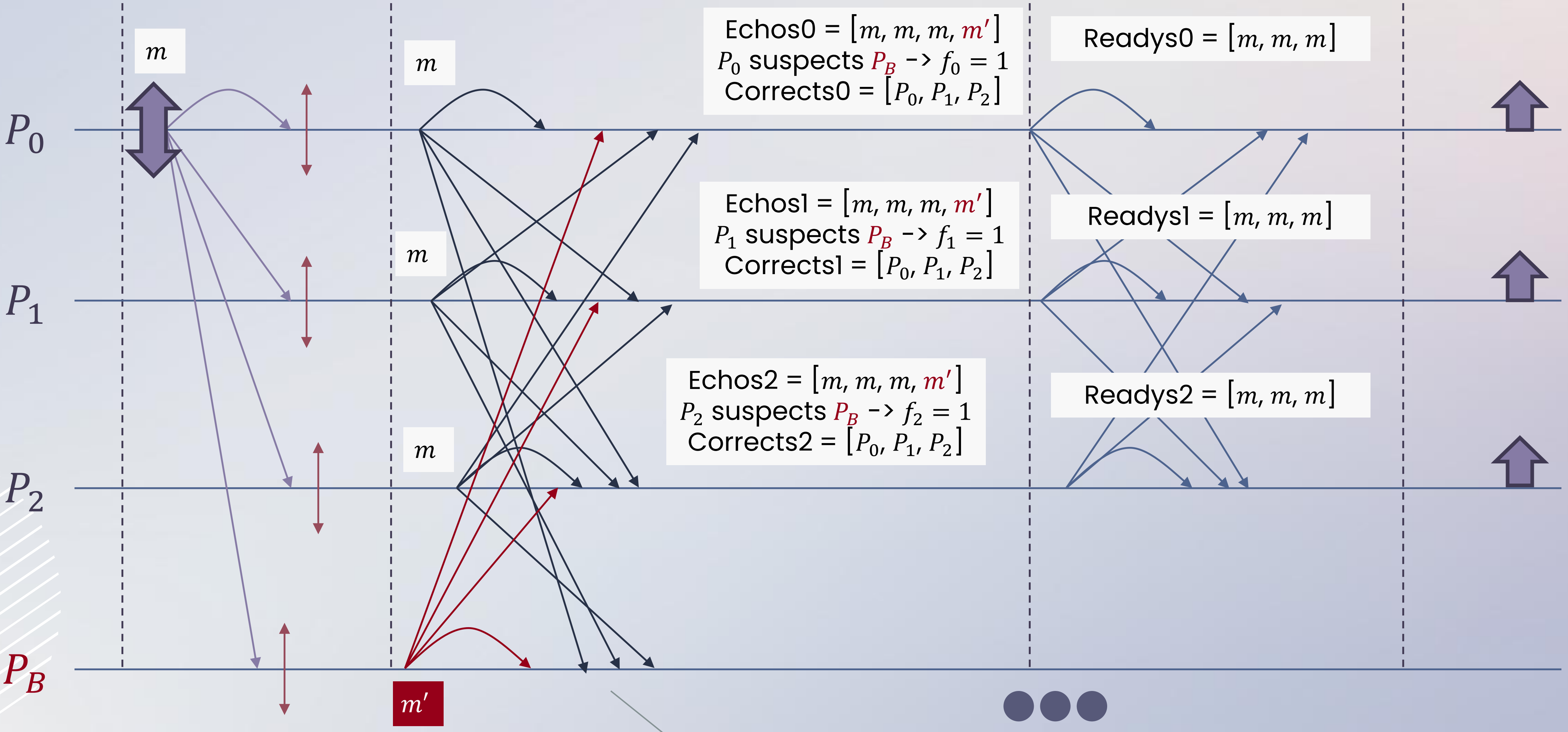
An Adaptive Solution(2)

- In the **ready set**, check whether happen that:

$$echos[p] \neq readys[p]$$

- If a process send a message for ready phase **different from which it sent in echo phase**, then this process is declared faulty
- Example:
  - $\Pi = \{P_0, P_1, P_2, P_3\}$, $s = P_0$
  - $P_3 : \{P_0 : m, P_1 : m, P_2 : m, P_3 : m\}_{Echo}$
  - $P_3 : \{P_0 : m, P_1 : m, P_2 : m', P_3 : m\}_{Ready}$
  - The process P_3 sets $f := f + 1$ and $faulty := faulty \cup \{P_2\}$

Simulation of the Adaptive Protocol



Echos0 = $[m, m, m, m']$
P_0 suspects P_B -> $f_0 = 1$
Corrects0 = $[P_0, P_1, P_2]$

Echos1 = $[m, m, m, m']$
P_1 suspects P_B -> $f_1 = 1$
Corrects1 = $[P_0, P_1, P_2]$

Echos2 = $[m, m, m, m']$
P_2 suspects P_B -> $f_2 = 1$
Corrects2 = $[P_0, P_1, P_2]$

Readys0 = $[m, m, m]$

Readys1 = $[m, m, m]$

Readys2 = $[m, m, m]$

Correctness of the Protocol

- The protocol always detects when there is heterogeneity in the echos or readys sets. But **how to identify the Byzantine process**?
- Example:
  - $\Pi = \{P_0, P_1, P_2, P_3\}, s = P_0$
  - $P_3 : \{P_0 : m, P_1 : m, P_2 : m', P_3 : m\}_{Echo}$
  - **Who is the Byzantine**? P_2 or P_0?
  - The process P_3 sets $f = 1$ but suspects both $suspected \coloneqq suspected \cup \{P_0, P_2\}$
- In case of uncertainty in echo phase, a **process exclude suspected from ready phase** except sender (initial trust)

What if the Sender is Byzantine?(1)

- Processes **check on ready set** if the following equation holds:

$$echos[p] = echos[s] = readys[s]$$

- **If no**, then the **sender is Byzantine** and **the process probably suspected a correct process**
- Example:
  - $\Pi = \{P_0, P_1, P_2, P_3\}, s = P_0$
  - $P_3 : \{P_0 : m, P_1 : m, P_2 : m', P_3 : m\}_{Echo}$
  - The process P_3 sets $f = 1$ but **suspects both** $suspected \coloneqq suspected \cup \{P_0, P_2\}$
  - $P_3 : \{P_0 : m', P_1 : m, P_3 : m\}_{Ready}$
  - P_3 **detects** $faulty \coloneqq faulty \cup \{P_0\}$ and $suspected \coloneqq suspected \setminus \{P_2\}$

What if the Sender is Byzantine?(2)

- Correct process suspected **may be stuck** due to symmetry
- Example:
  - $\Pi = \{P_0, P_1, P_2, P_3\}, s = P_0$
  - $P_2 : \{P_0 : m', P_1 : m, P_2 : m', P_3 : m\}_{Echo}$
  - No consistency message can be found, **symmetry** situation
  - P_2 is **stuck** because was misled by P_0 and **do not participate** in the ready phase
- When a process initiate its ready phase, it **start a timer** Δ in order to **wait some time** in case some process detects the sender as Byzantine

Recovery of Misled Process

- Whenever a correct process **detects the sender as Byzantine**, spread a **special message**: $ByzantineSender$
- The message **contains the echo set** of the detector process $echos\_p$
- When a process receives this message **check if**:

$$echos[s] \neq echos\_p[s]$$

- **If yes**, then spread its special message $ByzantineSender$
- When the process misled receives **enough special messages** $\#ByzantineSender \geq N - 2$, then it **can conclude that the sender was Byzantine** and **join** in the **ready phase**

Experiment 01

- $\Pi = \{P_0, P_1, P_2, P_3\}$
- $s = P_0$
- $Byzantine = P_0$
- P_1, P_2, P_3:
    - $\{P_0 : m', P_1 : m, P_2 : m, P_3 : m\}_{Echo}$
    - $\{P_1 : m, P_2 : m, P_3 : m\}_{Ready}$
    - $correct = \{P_1, P_2, P_3\}$
    - $faulty = \{P_0\}$
    - $f = 1$

Experiment 02

- $\Pi = \{P_0, P_1, P_2, P_3\}$
- $s = P_0$
- $Byzantine = P_3$
- P_0, P_1, P_2:
  - $\{P_0 : m, P_1 : m, P_2 : m, P_3 : m'\}_{Echo}$
  - $\{P_0 : m, P_1 : m, P_2 : m\}_{Ready}$
  - $correct = \{P_0, P_1, P_2\}$
  - $faulty = \{P_3\}$
  - $f = 1$

Experiment 03

- $\Pi = \{P_0, P_1, P_2, P_3\}$
- $s = P_0$
- $Byzantine = P_0$
- P_1, P_2, P_3:
  - $\{P_0 : m, P_1 : m, P_2 : m, P_3 : m'\}_{Echo}$
  - $\{P_0 : m, P_1 : m, P_2 : m\}_{Ready}$
  - $correct = \{P_1, P_2, P_3\}$
  - $faulty = \{P_0\}$
  - $f = 1$

Experiment 04

- $\Pi = \{P_0, P_1, P_2, P_3\}$
- $s = P_0$
- $Byzantine = P_3$
- P_0, P_1, P_2:
  - $\{P_0 : m, P_1 : m, P_2 : m, P_3 : m\}_{Echo}$
  - $\{P_0 : m, P_1 : m, P_2 : m, P_3 : m'\}_{Ready}$
  - $correct = \{P_0, P_1, P_2\}$
  - $faulty = \{P_3\}$
  - $f = 1$

Correctness of the Adaptive Solution

- The Adaptive Algorithm **executes correctly if**:

$$N > 3f$$

- Otherwise, **Byzantine process**(es) **may**:
  - Exclude some process from the communication (mislead)
  - Block the execution of some process (symmetry)
  - Change the message of the broadcast event (spoofing)
- If **sender is Byzantine**, then **the algorithm stops** because the sender of broadcast event was compromised
- Synchronous System to **avoid errors in detection**

Comments & Questions

- Open **Q&A**
- **Code** and **Experiments** in detail:
  https://github.com/GiuseppeDaidone/AdaptiveByzantineReliableBroadcast