Fault tolerance

Basic concept => Dependability

Dependability가 좋을수록 fault tolerance 하다.

1. availability -> 실제 동작시간, %로 표현

2. Reliability -> 시스템이 failure없이 동작하는 시간

MTTF : 동작을 안할 때까지 평균적으로 걸리는 시간

3. safety -> failure가 일어났을 때 심각한 위험을 초래하는지

4. maintainability -> 얼마나 빨리 복구할 수 있는지

MTTR : 복구하는데 걸리는 평균 시간

MTBF = MTTR + MTTF ( Mean Time Between Failures )

Availability(%) = MTTF / MTBF x 100

복구가 빨라지면 가용성이 높아진다.

Failure -> 우리가 원하는 동작 이외의 다른 모든 것

Error -> failure를 야기시킬 수 있는 시스템 상태

Fault -> error의 원인

Fault tolerance -> 시스템 상에 Failure를 야기시킬만한 여러 falut가 있음에도 정상적으로 서비스하는 것을 의미함

**[ Types of Fault ]**

* **Transient Fault** : 한번 fault 나고 안나는 거
* **Intermittent Fault** : 비주기적이라서 원인을 트래킹하기 어렵다.
* **Permanent Fault** : 발생된 이후로 계속 잘못된 값이 나오는 거

Failure models

1. Crash failures //동작을 정지한 것 reboot로 해결

2. Omission failures //들어온 요청에 대한 실패

- Receive omission : 어떤 값을 받고 처리를 안하는 거

- Send omission : 서버가 보냈는데 실제론 안 가는 거

3. Timing failures // 응답이 느리게 오면 failure로 가정할 수 있다.

4. Response failures// 응답 값은 왔지만 값이 틀린 경우

- Value failure : 요청은 제대로 왔지만 잘못된 값을 찾은 경우

- State transition failure : 잘못된 요청을 수신하고, 잘못된 상태로 실행되는 경우

5. Arbitrary(byzantine) failures : 이상한 값이 올 때도 있고 아닐 때도 있고 맞을 때도 있고 틀릴 때도 있고 애매하게 값이 와서 가장 위험한 상황이다.

Redundancy를 사용하여 fault tolerance를 한다.

-> Information redundancy 주고 받는 정보 속에 중복된 데이터를 넣어 복구 시킨다. ( FEC )

-> Time redundancy 동일한 내용의 정보를 다시 보낸다.(retransmission)

-> Physical redundancy 동일한 작업을 수행하는 sw나 hw를 여러 개 중복시킨다

->Triple modular redundancy .

Process Resilience

***Resilience : 회복 탄련성***

프로세스들을 그룹 내에서 복제시켜 프로세스의 손실 혹은 실패를 보호하고 회복한다.

* **Group Organization**
  + **Flat Group** : Coordinator가 존재하지 않고 모두 동일한 작업을 수행하게 된다.

동일한 작업을 모두 수행하기 때문에 하나가 망가지더라도 전체 그룹에 큰 영향 끼치지 않는다.

하지만, 일 처리 효율성은 떨어진다.

* + **Hierarchical Group** : **Coordinator**가 존재하여 프로세스들간의 업무 배정을 진행한다.

동일한 작업을 모두 똑같이 수행하는 것이 아니라 업무를 배정하기 때문에 효율성은 높아진다.

그렇지만, 그에 비해 Coordinator가 다운될 시 전체 그룹이 다운될 수 있기에 위험성이 높다.

(single point failure)

Coordinator 실패시 Election을 통해 다시 선정한다. - Bully & Ring

Groups and Failure Masking

-> k fault tolerant 그룹은 k개의 failure에 대해 masking(숨기는 거)이 가능하다.

어느 정도 멤버가 있어야 가능한다.

- crash failure semantics => k + 1 members

- arbitrary failure semantics / group output defined by voting => 3k+1 members

( byzantine failure tolerance )

Byzantine General’s Problem

**IC1**: All loyal generals get same result

커맨더가 충직하지 않더라도 loyal 장군들은 같은 값을 가지고 있어야 한다.

**IC2**: If commander is loyal, all loyal generals follow his choice

만약 commander가 충직하면 나머지 충직한 장군들은 그 의견을 따라야한다.

No solution will work unless there are more than **2/3** loyal ones