The Primary-Backup Approach

Chapter 8
Chapter 8 : Contents

- Introduction
- The Specification of P-B
- A Simple P-B protocol
- Lower Bounds of P-B approach
  - Degree of Replication
  - Blocking Time
  - Failover Time
Introduction

• Primary and Backups
  – Clients make requests only to the primary
  – Backups ignore clients’ requests
  – If the primary fails, then one of the backups takes over (failover)
Three Key Metrics of P-B system

• Degree of replication
  – less, better

• Blocking time
  – the lag between request receipt ~ worst response
  – normal processing latency

• Failover time
  – the worst-case period during which requests can be lost
Specification of Primary-Backup

• 4 Properties to be a P-B protocol
  – PB1: only one primary at any time
  – PB2: every client maintains a server $Dest_i$
  – PB3: backup ignores client’s requests
  – PB4: $\exists k, \Delta$ such that service behaves a single $(k, \Delta)$-bofo server
    • bonded outage, finitely often
    • only $k$ outage periods sums up to $\Delta$ period
A Simple P-B Protocol

- Primary processes a request
  - process => send to p2 => respond
  - send ‘I’m alive’ to backup every $\tau$
  - what if p2 do not receive the request?
A Simple P-B Protocol

- Performance Analysis
- Failover time
  - $2\delta + \tau$
- Degree of Replication
  - at least 1 backup
- No blocking Time
  - $(1, \tau + 4\delta)$-bofo

Distributed Computing Systems
A Simple P-B Protocol

- Performance Analysis

- Failover time
  - $2\delta + \tau$

- Degree of Replication
  - at least 1 backup

- No blocking Time

- $(1, \tau+4\delta)$-bofo

P1 may have failed $(t-\delta, t)$
Lower Bounds of PB approach

• Degree of Replication
  – crash : n > f
  – crash + link : n > f + 1
  – receive-omission : n > \lfloor 3f/2 \rfloor
  – send-omission : n > f (obvious)
  – general-omission : n > 2f
Lower Bounds on Replication

- Crash + Link Failures

- If \( n = f \),
  - All \( f \) links between A and B fails
  - A is primary
  - one of B becomes also primary

- Thus \( n > f \)
Lower Bounds on Replication

- Receive-Omission

\[ \left\lfloor \frac{3f}{2} \right\rfloor \]

- If \( n = \left\lfloor \frac{3f}{2} \right\rfloor \),
  - A&C crash => B primary
  - B&C crash => A primary
  - A, B respectively fails as receive-omission, there will be primary in both A and B
Lower Bounds on Replication

- General-Omission

- If \( n = 2f \),
  - Each partition will have a primary

\[ \begin{align*}
  &\text{A} \\
  &\text{B}
\end{align*} \]
Non-Blocking Protocols

• Protocol for crash failure
  receive request \( (n > f) \)
  process it
  broadcast to all backups
  send response

• Protocol for crash + link failure
  : \( (n > f+1) \)
  echo broadcasts to other backups
  (at least 1 path exists between primary & backup)
  :

Distributed Computing Systems
Lower Bounds on Blocking Time

- **δ-Blocking Protocol**
  - \( f = 1 \)

- **2δ-Blocking Protocol**
  - \( f > 1 \)
Lower Bounds on Failover Time

- **Failover times**
  - Crash : $f\delta$
  - become primary in predetermined order
  - 1st backup timeouts after $\tau+\delta$, 2nd after $\tau+2\delta$...
  - $\tau = 0$ to compute lower bounds

- **link failure** : (longer forwarding path)
  - primary broadcasts, backups echo ($n > f+1$)
  - 1st backup timeouts after $\tau+2\delta$, 2nd after $\tau+4\delta$...
  - failover time : $f(2\delta+\tau)$

- **Simple property**
  - PB5 : A correct primary remains so until there is a failure

Distributed Computing Systems 15
Protocols for more severe failures

• Translation technique
  – translate omission failure to crash using 2 phase
    when p sends a message, it broadcasts to all
    processes and everybody echoes
    if fewer than n-t echoes, detects failure and crash

  – translate omission failure to arbitrary failure
    using Reliable Broadcast + Validation
Alsberg & Day Protocol

- Earliest primary-backup protocol
- For two servers and tolerates a single crash
  - Clients makes requests to either servers
  - Primary processes, propagates to backup, and blocks, backup processes, responds to client, and sends ack to primary which unblocks.
  - Backup forwards to primary, primary processes, replys and sends update msg to backup, backup updates and discards the request.
Tandem protocol

• For single crash+link failure
• System Configuration
  – multiple nodes connected by a network
  – each node consists of multiple processor and IOP interconnected by redundant buses
  – each processor can support multiple processes
• Process-pairs
  – replicate each process on two different processors in the node as primary and backup
  – primary sends a state update using one bus, backup acks and primary reply (periodically sends “I’m alive” through 2 buses)
  – if no ack from backup, resend using redundant bus
  – if no msg for sometime, backup takes over
HA-NFS

- Tolerates a single crash by using 2 servers
  - servers are connected to a dual-ported disk
  - disk failures are masked by mirroring
  - link failures are tolerated by replicating network between clients and servers
  - primary does not inform the backup of the state update (dual-ported disk)
  - backup takes over if no I’m alive message + it cannot communicate with the primary through disk
Non-blocking protocol tolerating receive omission failure

• Requires 2f+1 servers

• Primary
  – upon receiving and processing a request, propagates the state update to all the backups and responds to the client
  – backups update according to the message
  – backups may experience receive-omission
    => needs failure detection (n > 2f )

• Failure detection
  – periodically exchange *I’m alive* message

• Problems
  – O(n^2) messages to detect failure => network congestion