Leases in Distributed Systems

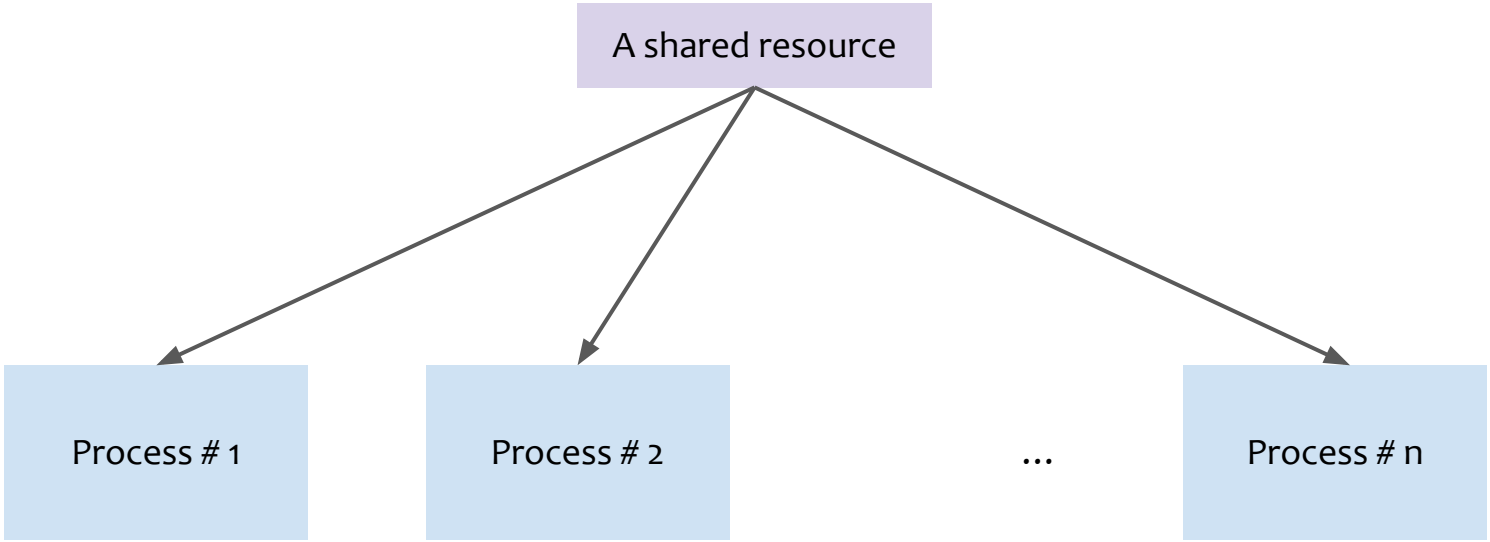
Prof. Pramod Bhatotia
Chair of Decentralized Systems Engineering (DSE)
Department of Computer Science
https://dse.in.tum.de/

WS 2022/23

Lease

A fundamental abstraction for scalable resource management

A typical problem



How to access a shared resource in an exclusive manner?

Locking as a panacea?

```
acquireLock(shared_resource);
//access the shared resource
releaseLock(shared_resource);
```

- A process can crash

  OR

- They can have network failure

The problem: A process can hold the resource indefinitely!

Lease: An alternative fault-tolerant abstraction

- A lease is a contract that gives its holder specified rights over a resource for a limited period of time
  - A process can ask for a lease for a limited period of time, after which it expires
  - The process can renew the lease before it expires if it wants to extend the access

Lease variants

- **Read lease** allows client to cache clean data
  - **Guarantee:** no other client is modifying data

- **Write lease** allows safe delayed writes
  - Client can locally modify the data, and then writes in a batch
  - **Guarantee:** no other client has data caches

Using leases

- Client requests a lease
  - Specifies the resource id
- Server determines if lease can be granted
  - Read leases may be granted concurrently
  - Write leases are granted exclusively
- If conflict exists, server may send eviction notices
  - Evicted write lease must write back
  - Evicted read leases must flush/disable caching
  - Client acknowledges when completed

Bounded lease term simplifies recovery

- Before lease expires, client must renew lease

- Client fails while holding a lease?
  - Server waits until the lease expires, then unilaterally reclaims
  - If client fails during eviction, server waits then reclaims

- Server fails while leases outstanding? On recovery:
  - Wait lease period + clock skew before issuing new leases

Caveat

- Leases depend on well-behaved clocks
  - If the clock speeds are mismatched between the lease grantor and the lease holder, it can lead to inconsistencies

More example use-cases

- Google's Chubby and Apache Kafka: Co-ordination services
  - https://research.google/pubs/pub27897/
  - https://zookeeper.apache.org/
- Apache Kafka: an event streaming platform
  - https://kafka.apache.org/
- Google’s Thialfi – a client notification service
  - https://research.google/pubs/pub37474/
- DHCP protocol
  - https://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol
- Google’s Slicer: Auto-sharding
  - https://research.google/pubs/pub46921/

References

- **Seminal paper:**
  - https://web.stanford.edu/class/cs240/readings/89-leases.pdf

Leases: An Efficient Fault-Tolerant Mechanism for Distributed File Cache Consistency

Cary G. Gray and David R. Cheriton
Computer Science Department
Stanford University