**Overview**

- Distributed storage often shares cluster machines.
  - E.g., within data-intensive computing frameworks.
- Want ability to grow/shrink server set elastically.
- Adapting to demand.
- Releasing unneeded servers for other activities.
- Traditional distributed storage not elastic.
- Primary/non-primary data layouts allow this.
  - One copy of all data on primaries.
  - Can ensure availability with subset of servers.
  - Replicas stored on non-primaries.
  - Can elastically activate/release these servers.
- Goal: A storage system that can:
  - Deactivate/reactivate servers quickly to save machine hours.
  - But still maintain high performance at the same time.

**Equal Work Data Layout**

- P primaries and (N-P) non-primaries.
- Equal work arrangement on non-primaries.
  - Number the servers, starting with the P primaries.
- Store \( \geq B/X \) blocks on non-primary server \( X \).
- Guarantees equal distribution of read work.
  - Even when active set grows or shrinks.
- Number (P) of primaries creates tradeoff.
  - Small P maximizes elasticity.
  - Small P creates a write bottleneck.
- Offloading removes the tradeoff.
  - Offload reads from primaries, when possible.
  - Offload writes, when necessary, to offload set.
- Explicit offload set retains agility.

**Read and Write Data Offloading**

- Number (P) of primaries creates tradeoff.
- Small P maximizes elasticity.
- Small P creates a write bottleneck.
- Offloading removes the tradeoff.
- Offload reads from primaries, when possible.
- Offload writes, when necessary, to offload set.
- Explicit offload set retains agility.

**Policy Analysis with Industrial Traces**

- Real-world traces reveal great potential for machine hour saving.
- JackRabbit wins over state-of-art elastic storage systems like Rabbit and Sierra.
- JackRabbit significantly reduces machine hour usage and data migration.

**JackRabbit Performance**

- JackRabbit implements equal work and offloading.
- Implemented as modified HDFS.
- Read throughput equal to or better than HDFS.
- Write throughput scales with offload set.
- Minimize cleanup overhead.

**Other Layout Features**

- Fault-tolerant elasticity, via gearing.
  - Organize each primary's secondary replicas.
  - Failure of a primary then doesn't remove elasticity.
- Multi-volume data layout.
  - Have each volume use distinct primaries.
  - One volume's primaries are others' non-primaries.
  - Allows small P without underutilized capacity.