Be Fast, Cheap and in Control with SwitchKV

Xiaozhou Li
Raghav Sethi
Michael Kaminsky
David G. Andersen
Michael J. Freedman
Fast and cost-effective key-value store

• **Target:** cluster-level storage for large, active data sets
  – Small key-value items, persistent, strongly consistent

• **Goal:** meet the service level objectives (SLOs)
  – Aggregate throughput and tail latency

• **Fast SSDs are opening up new points in the design space**
  – Emerging hardware and software technology
  – Can meet the SLOs of many cloud services *cost-effectively*
Scale out SSD-based cluster

- Meet the SLOs without substantial over-provisioning
Scale out SSD-based cluster

• Meet the SLOs without substantial over-provisioning
  – under widely varying and rapidly changing workloads.
Key challenge: dynamic load balancing

- Load imbalance hurt performance
  - Lower throughput
  - Higher (tail) latency

- Existing methods (e.g., data migration) have limitations
  - system overhead
  - consistency challenge

- Fast, small cache can ensure effective load balance
  - Only need to cache the $O(n \log n)$ hottest items, $n$ is the total number of backend nodes [Fan, SOCC’11]
SwitchKV: heterogeneous key-value storage cluster

• Large scale SSD-based backend servers
  – Cost-effective but resource-constrained
  – Provisioned for the performance goals

• Specially-configured high-performance node
  – Fast, small in-memory cache
Traditional systems: cache must process all queries

- High system overhead when cache hit ratio is low
  - Throughput is bounded by the cache
  - High latency for queries for uncached keys

Look-aside architecture

Look-through architecture
SwitchKV: effective content based routing

• Exploit SDN and deeply optimized switch hardware
  – Encode key information in the packet MAC header
  – Install exact match rules for all cached keys
  – Switches forward requests directly to the right nodes
How to update the cache?

- Backends add each recently-visited key to the cache?
  - Works fine when the cache hit ratio is high
  - What if backends are responsible for most queries?
    - Unnecessary cache churn
    - High bandwidth and computation overhead
    - Switch rule update rate is limited
SwitchKV: minimize unnecessary cache churn

Controller -> Cache

Cache <-> Backend

- Top-k <key, load> list (periodic)
- fetch request <key>
- fetch request <key, value>

Controller

- switch rule update

- bursty hot <key, value> (instant)
Conclusion

• SwitchKV: load-balanced cluster-level key-value store
  – Load balancing guaranteed by fast, small cache
  – Efficient content-based routing
  – Reacts quickly to workload changes with hybrid cache updates

• Meet the SLOs more efficiently than traditional systems

• Checkout our NSDI paper this March 😊