Understanding how the network impacts performance and energy-efficiency in the RAMCloud storage system

Yacine Taleb, Shadi Ibrahim, Gabriel Antoniu, Toni Cortes

Understanding how the network impacts performance and energy-efficiency in the RAMCloud storage system

Yacine Taleb
Inria Rennes
Bretagne-Atlantique
Rennes, Britanny, France
yacine.taleb@inria.fr

Shadi Ibrahim
Inria Rennes
Bretagne-Atlantique
Rennes, Britanny, France
shadi.ibrahim@inria.fr

Gabriel Antoniu
Inria Rennes
Bretagne-Atlantique
Rennes, Britanny, France
gabriel.antoniu@inria.fr

Toni Cortes
Barcelona Super Computing Center
Universitat Politècnica de Catalunya
Barcelona, Spain
toni.cortes@bsc.es

ABSTRACT

In-memory storage systems emerged as a de-facto building block for today's large scale Web architectures and Big Data processing frameworks. Many research and engineering efforts have been dedicated to improve their performance and memory efficiency. More recently, such systems can leverage high-performance networks, e.g., Infiniband. To be able to leverage these systems it is essential to understand the trade-offs induced by the use of high-performance networks. This paper reports on work in progress aiming to contribute to a better understanding of the main factors impacting performance of in-memory storage systems. Through a study carried on RAMCloud, we focus on two settings: 1) clients are collocated within the same network as the storage servers (with Infiniband interconnects); 2) clients access the servers from a remote network, through TCP/IP. We compare and discuss aspects related to scalability and power consumption for these two scenarios which correspond to different deployment models for applications making use of in-memory cloud storage systems.

Keywords

In-memory, Cloud, Performance Measurement, Energy Efficiency

1. INTRODUCTION

Nowadays, the size of data generated by digital media, social networks and scientific instruments is increasing at an extreme rate. Moreover, recently new types of data-intensive applications emerged. In this type of applications data is shared and updated by many users (e.g., photos tagging, on-line gaming, etc).

For these applications, low response time and high throughput are essential. To satisfy such needs, large-scale Web providers, such as Facebook [14] or Twitter [9] are placing most accessed data in DRAM to provide low response times. As main memory capacity is increasing, a large number of in-memory storage systems have emerged from both industry and research. These systems mostly focus on performance [7, 3, 12, 8] or features such as fault-tolerance [15], or memory efficiency [18].

Performance and memory efficiency are important for these systems, but energy efficiency has been neglected. Given that main memories of DRAM-based servers consume between 25% up to 40% of the total energy of the servers [10], it is important to study the energy efficiency of in-memory storage systems (e.g., the trade-off between performance and energy consumption). Especially as energy has become a major issue in today's infrastructures [1].

There are many factors that can impact performance and energy efficiency, e.g, the network. Recent in-memory storage systems can leverage high performance networking technologies such as Infiniband networks [15, 12], and special features such as RDMAs [7, 13, 20].

While many works suppose that the clients can leverage the same network technology as the storage system, it is not always the case [13, 15, 7, 12, 20]. For example a common case where clients share the same network as the storage system is a HPC setup. For large scale Web applications, e.g., social networking, the users access the system through common TCP sockets.

Therefore, understanding the role of the networking technologies on performance and energy efficiency is important. It can help identifying the main performance bottlenecks and factors contributing to energy inefficiency.

The objective of our work is to give a clear picture of the performance and energy-efficiency of a representative in-memory storage system, namely RAMCloud [15, 18]. Through an experimental study carried on GRID'5000 [2] we identify and discuss the main trade-offs arising in both cases where clients are sharing the same network as the storage system

and when they are not. Our study confirmed foreseen results such as better performance when clients are collocated with the storage system. On the other hand, we found that when clients are not collocated within the same network RAMCloud is not energy efficient. When investigating the main cause we found that it stems from the fact that using TCP/IP transport in RAMCloud leads to longer CPU usage which leads to more power consumption for longer periods.
The remainder of this paper is organized as follows: Section 2 we discuss why we chose RAMCloud and gives background about its architecture. Section 3 presents our experimental methodology and describes the benchmark and platform used. In Section 3.4 we present the results of our experiments and discuss the main finding. Section 4 presents related work. Finally, Section 5 concludes this work and gives insight about future directions.

2. BACKGROUND

In the past two decades a lot of efforts have been dedicated to improve the performance of storage systems due to the increasing demand of large Web applications and Big Data applications.
Traditionally the hard-disk was known as the bottleneck in clustered storage systems and distributed processing frameworks [11]. Then, caching and in-memory storage systems emerged as solutions to rely on main memories speed. The bottleneck was not anymore disk but CPU and network [16]. As a consequence, recent in-memory storage systems can leverage high speed networks such as Infiniband. However, it is still not clear how these systems perform when such high speed networks are not available for client-to-server communication.

2.1 A representative system: RAMCloud

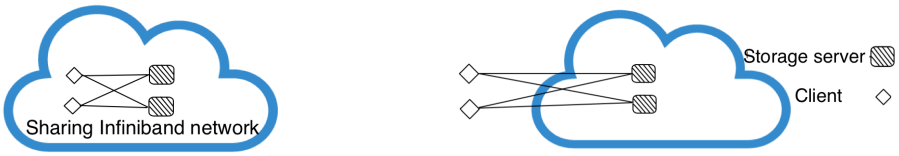
In contrast with most of the recent in-memory storage systems, RAMCloud offers a complete storage system with availability and durability guarantees. Most of the recent systems focus on a particular features, e.g., Pilaf focuses on low latency through using RDMAs [13] but has no guarantees for data durability. Mica [12] focuses on high performance but does not provide efficient-memory usage. MemC3 [8] is an evolved version of Memcached, but it does not provide any durability or availability guarantees. The closest system to provide all these features to be found in the literature is FaRM [7], unfortunately it is neither open-source nor publicly available.
The importance of having all these features together is to have a clear picture of the performance and energy efficiency of the storage system as a whole. Although in this work we focus on the networking part of the system, we will study the impact of all these design principles in terms of performance and energy-efficiency in future work.

2.2 The RAMCloud storage system

RAMCloud is an in-memory key-value store. The main claims of RAMCloud are low-latency, fast-crash recovery, and efficient memory usage. Operations on small objects can be achieved in few microseconds by using high performance networks (e.g., Infiniband). Fast-crash recovery is achieved by randomly replication data to as many backups as possible in the cluster then reconstructing data by exploiting in parallel backups and recovery servers. Memory efficiency is achieved through using DRAM as a log-structured memory and through efficient memory cleaning. Next we present the main concepts and mechanisms that are essential background to understand our study.
Architecture A RAMCloud's cluster consists of three entities : a coordinator maintaining meta-data information about storage servers, backup servers, and data location; a set of storage servers that expose their DRAM for the clients as storage space; and backups that will store replicated data in their DRAM temporarily and persist it to disk asynchronously. Usually, storage servers and backups are collocated. Thus a physical machine will run both storage and backup services at the same time.
Data structures Data in RAMCloud is stored in a set of tables. Each table can span multiple storage servers. A table is partitioned into a set of *tablets*. A uniform hash function distributes evenly the objects across tablets.
A server in RAMCloud uses a log-structured memory to store its data. It uses a hash-table to index data. This log-structured approach, coupled with a two level cleaning approach, enable RAMCloud to use DRAM efficiently.
The log-structured memory of each server is divided into 8MB segments. A server stores data in an append-only fashion. Thus, to free unused space a log-cleaner is called whenever a server reaches a certain memory utilization threshold. The cleaner copies a segment's live data into the free space (still available in DRAM) and removes the old segment.
As the cleaner is called more often in memory, when data on disk grows larger than a threshold the two-level cleaning starts, this time cleaning data in both memory and disk at the same time.
Data durability: It is one of the most important aspects of RAMCloud, as it has impacted most of its design. In RAMCloud data is present all time in DRAM. However, durability is guaranteed by replicating data to remote disks. More precisely, whenever a storage server receives a write request, it appends the object into its latest free segment, and forwards a replication request to the backup servers randomly chosen for that segment. The server waits all acknowledgements from backup servers to answer client's update request. Backup servers will keep a copy of this segment in DRAM until it fills, then flush the segment to disk and remove it from DRAM.
For each new segment, a random backup in the cluster is chosen. This enables RAMCloud to harness large-scale to enable fast crash recovery. Moreover, when a server crashes, multiples servers will be involved in the crash recovery, each one of them taking responsibility of a set of segments to ensure even re-distribution of the recovered data.

2.3 Location of the clients: Why is it important?

As a motivating example, Figure 1 shows the two possible ways of using RAMCloud. In Figure 1a the clients are sharing the same datacenter network as the storage system. In this case the clients can take full advantage of the storage system and benefit from high speed networks (when available). On the other hand, a second use case is displayed in Figure 1b, where the clients might use traditional TCP/IP networking stack to access the storage system. In this case clients will have slower access to the system, and more importantly it is not clear whether they will benefit or not from



(a) Clients and servers share the same network

(b) Clients access servers from outside the datacenter

Figure 1: The two possible scenarios for accessing RAMCloud. Figure 1a refers to the case where the clients also use Infiniband transport. Figure 1b represents the case where the clients are not collocated within the same datacenter with the storage servers. We refer to this case in RAMCloud as the one where the client uses TCP/IP

the speed of the in-memory storage.
Through an experimental study, we attempt to answer these questions by reproducing both scenarios displayed in Figure 1. In Section 3.4 we present insights about the performance and energy-efficiency of RAMCloud in both cases.

3. EXPERIMENTAL EVALUATION

3.1 Benchmark

We used the industry standard Yahoo! Cloud Serving Benchmark (YCSB) benchmarking framework [6]. YCSB is an open and extensible framework that allows to mimic real-world workloads such as large scale web applications, to benchmark key-value store systems. YCSB supports a large number of databases and key-value stores, which is convenient for comparing results with different systems.
Basically, one needs to fill the data-store with a YCSB client. It is possible to specify the request distribution, i.e., uniform, zipfian, etc. Executing the workload consists of running clients with a given workload specification, i.e., number of requests, distribution of requests, number of threads per clients, etc.

3.2 Platform

The experiments were performed on the Grid'5000 [2] testbed. The Grid'5000 platform provides researchers with an infrastructure for large-scale experiments. It includes 10 geographical sites spread across French territory except 1 located in Luxembourg.
We relied on Nancy's site nodes to carry our experiments. More specifically, the nodes we used have 1 CPU Intel Xeon X3440, 4 cores/CPU, 16GB RAM, and 298GB HDD. Additionally each one has an Infiniband-20G and a Gigabit Ethernet card.
We have chosen these nodes as they offer capability to monitor power consumption: 40 of these nodes are equipped with PDUs which allow to retrieve power consumption through an SNMP request.
Throughout all our experiments we make sure to reserve the whole cluster which consists of 133 nodes, to avoid any interference with other users of the platform. We dedicate the 40 nodes equipped with PDUs to run RAMCloud's cluster, i.e., master and backup services. We run YCSB clients on the rest of the nodes to avoid any interference of any sort.

3.3 Experiments' Configuration

We deployed RAMCloud on the testbed described in 3. Only the 40 nodes having power measurement capabilities were used to run RAMCloud, 90 other nodes were used as client machines, and 1 node as a coordinator. It is important to note that each client runs on a single machine, this helps us avoid interferences of any sort. Each node ran as server and backup at the same time. We have fixed the memory used by a RAMCloud server to 10GB and the available disk space to 80GB. We have fixed the memory size to an acceptable limit to carry the whole data in the cluster. Before running each benchmark, we pre-load 100K records of 1KB in the cluster. Running the benchmark consists of launching simultaneously one instance of a YCSB client on each client node. Each client issues 100K requests, which corresponds to the total number of records. Having an increasing number of requests with the number of clients enabled us to study concurrency impact's on RAMCloud's scalability.
Having each client generate 100K requests results in having 1M requests with 10 clients for example, and 9M requests with 90 clients, which corresponds to 8.58GB of data requested per run.
In our figures, each dot correspond to an average of 5 runs with the corresponding error bars. When changing the cluster configuration (i.e., number of RAMCloud servers), we remove all data stored on servers as well as in backups, then we restart the RAMCloud servers and backups on the whole cluster to avoid any interference with prior experiments.
We configured RAMCloud with the option *ServerSpan* equal to the size of the cluster. As RAMCloud does not have a smart data distribution strategy, this option is used to manually decide the distribution of data across the servers. Moreover, we used uniform data distribution in YCSB clients to insert and request data. At the end, all the data is evenly distributed across the servers and each object is requested at the same frequency. This allows us to avoid hot spots and thus have uniform load distribution.

3.4 Results

3.4.1 Performance and scalability

We present the results when running RAMCloud with both TCP and Infiniband configurations. We measured throughput and energy consumption. We varied the cluster size from 10 to 40 nodes and the clients from 10 to 90 clients with a read-only workload.
Figure 2 shows the aggregated throughput for a 10 node cluster with both Infiniband and TCP. Without surprise there is a big gap in the total throughput achieved by the Infiniband enabled cluster and the TCP one. When running 10 clients Infiniband cluster achieves 7.38x the throughput of TCP cluster. Whenever increasing the number of clients the gap widens and reaches 11.40x more throughput for Infiniband cluster. What stands out here is the limited scalability of

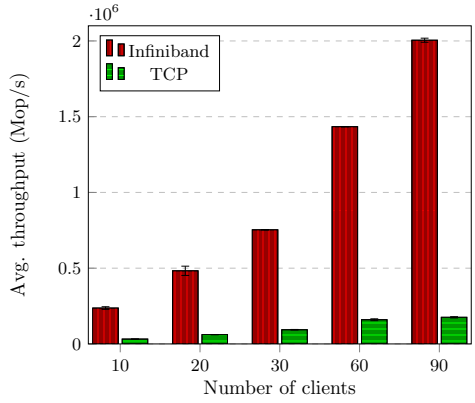


Figure 2: Total aggregated throughput as a factor of clients number with a fixed cluster size of 10 nodes.

TCP, for example between 60 and 90 clients while Infiniband increases by a factor of 1.39x TCP only increases by 1.10x from 60 to 90 clients. This suggests that since the transport protocol is slowing down the operations, RAMCloud nodes are subject to more concurrency, and thus scalability might suffer.

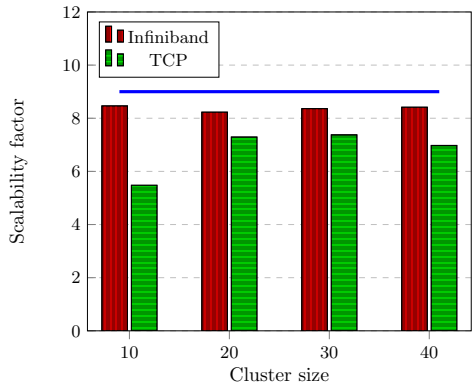


Figure 3: Throughput scalability factor as a function of cluster's size and when running 90 clients. The blue line represents the expected increase in throughput according to the baseline of 10 clients.

To further explore that behaviour we plot in Figure 3 the scalability of each of the scenarios, i.e., varying the cluster size from 10 to 40 nodes and we fix the clients to 90. The figure represents the ratio of throughput increase when taking 10 clients as a baseline. When clients are collocated with RAMCloud, it can achieve linear scalability as we already showed in the previous section, even under high concurrent accesses. On the other hand, we can witness the difference widening between TCP scalability and the expected ratio. For the cluster of 10 nodes with TCP the scalability factor is 5.4 while the expected scalability factor is 9. This difference reduced when increasing the cluster size, for example it 6.9 for 20 nodes and 7.3 for 30 nodes. This confirms our observation about RAMCloud being more fragile facing high workloads from clients from outside compared to when they are collocated with the same network. We suspect this is due to lesser load on resources as we increase cluster size.

As expected, when clients are collocated on the same network as RAMCloud it can achieve more throughput compared to when clients access the system from a different network. In the same lines, when clients are collocated with RAMCloud it achieves better scalability.

3.4.2 *Energy efficiency*

Figure 4 represents the average power consumption of the cluster for the same experiment described above. For the Infiniband cluster we can see a stable power consumption of 94Watts up to 60 clients, while it increases a little bit when going up to 90 clients. More likely, this is the point where the cluster starts demanding more resources to cope with the load, which matches our observation from section 1. If we look at the TCP cluster power consumption the results are more surprising. First we see that the average power consumption is increasing constantly from 97Watts for 10 clients up to 109Watts for 90 clients. Moreover, in all cases it is higher than the Infiniband cluster average power consumption.

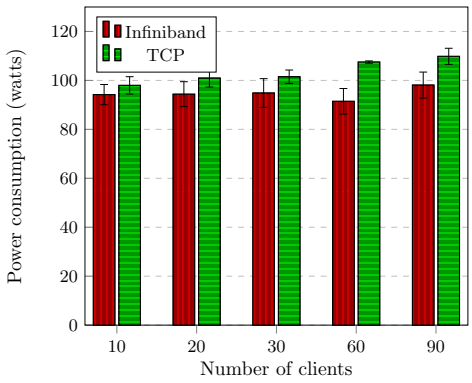


Figure 4: Average power consumption per node in Watts. The cluster size is fixed to 10

Again, these results were counter-intuitive since one expects more power consumption from nodes delivering more throughput. To explain that behaviour we plot Figure 5 which represents the latency of a single operation for the same scenario, i.e., 10 servers running up to 90 clients. When looking into Infiniband's cluster latency one can see a stable latency around 40 μs. For TCP cluster the latency is stable around 315 μs up to 60 clients. It reaches more than 512 μs for 90 clients.

Therefore, it is easier to see that a node running TCP transport protocol will be busier and for a longer time compared to a node using Infiniband. When we looked at the average CPU usages, we could confirm this assumption, since under when clients are using TCP all nodes have a higher CPU usage than when collocated with RAMCloud.

> Surprisingly, when running RAMCloud with TCP transport it consumes more power than using Infiniband. We traced back this issue to the additional latency generated when using TCP. The additional latency leads to longer CPU occupation periods, which

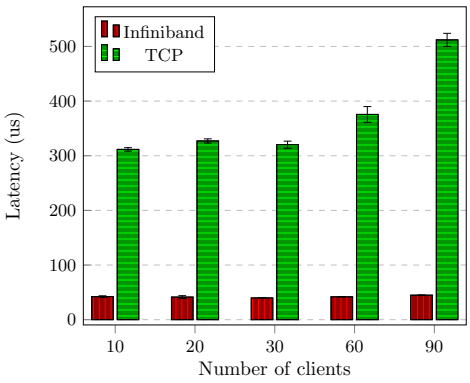


Figure 5: The latency per single operation when running read-only workload with a cluster of 10 nodes both with TCP and Infiniband

ends up in additional energy consumption.

4. RELATED WORK

Recently, in-memory storage systems have become popular with the decrease of per-byte DRAM price. Researchers have focused mainly on improving the performance and reliability of in-memory storage systems. Although there have been work trying to improve the energy efficiency of disk-based storage systems, very few work has been dedicated to explore the energy efficiency of DRAM based storage systems. On the one hand, energy consumption is a major issue in nowadays computer systems. On the other hand, main memory is contributing to a significant part of the total energy consumption of a modern DRAM-based server. More than ever it became vital to investigate sources of energy inefficiency in in-memory storage systems to better tackle this issue.

In-memory storage systems More and more companies are adopting in-memory storage systems. *Facebook* leveraged Memcached to build a distributed in-memory key-value store. Twitter used *Redis* and scaled it to 105TB of RAM. Alternatively, a lot a academic work has been proposed pushing the limits of performance of storage systems to a next level. As an example *MemC3* [8] is an enhancement of Memcached. Through a set of engineering and algorithmic improvements, it outperformed the original Memcached by 3x in terms of throughput. *Mica* [12] is an in-memory key-value store that takes advantage of RDMAs (Remote Direct Memory Access) and better parallelism handling. Similarly, *FaRM* [7] is a distributed in-memory storage system based on RDMAs and proposes transactional support.

Evaluating storage systems performance Many studies have been conducted to characterize the performance of storage systems and their workloads. In [5] a deep analysis is made on traces from Facebook's Memcached deployment and gives insight about the characteristics of a large scale caching workload. In another dimension, the work in [17] proposes to study the throughput of six storage systems. However, the study does not give insights about the different factors and design principles contributing positively or negatively to performance.

Energy efficiency Many systems have been proposed to tackle the energy efficiency issue in storage systems. For instance *Rabbit* [4] is a distributed file-system built on top of *HDFS* (Hadoop File System) that tries to turn turn off the largest possible subset of servers in a cluster to save energy with the least performance decrease. It arranges the data in a way to minimize the number of nodes to be turned-on in case of primary replica failure. In a similar way *Sierra* [19] is a distributed object store that aims at power-proportionality. Through a three-way replication, it allows servers to be powered-down or put in stand-by in times of low I/O activity. This is made possible through a predictive model that observes daily I/O activity and schedules power-up or power-down operations.

To the best of our knowledge, this work is the first to propose a comprehensive study on the network's impact on energy efficiency and performance in an in-memory storage system.

5. CONCLUSION AND FUTURE WORK

In this paper we presented our efforts in understanding the main factors impacting performance and energy efficiency in in-memory storage system. We zoomed into the networking technology impact on the RAMCloud storage system.

We confirmed foreseen results such as the scalability of RAMCloud's throughput when using Infiniband. However, we discovered that using RAMCloud with TCP does not scale as well as with Infiniband. This phenomena becomes dramatic at high loads. Moreover, we discovered that using TCP leads to more energy consumption compared to using Infiniband. When investigating the issue, we remarked that it was due to higher CPU occupation times whenever running with TCP.

As future work we plan to complete the picture of the main factors impacting performance and energy efficiency. To do so we plan to break down the design principles of RAMCloud such as the polling mechanism, log-structured memory, replication, crash-recovery mechanism, and study the impact of each of them on the performance and energy efficiency. Doing so can help system designer to build more energy-aware systems in the future. It can also open new research opportunities in investigating the trade-offs between energy efficiency and performance in in-memory storage systems.

Our ultimate goal would be to model the performance and energy consumption for in-memory storage systems. While this is a very hard task, it can have a huge impact as it can enable system designers to be aware of the impact of each feature in their system on the performance and energy consumption. Moreover, it can help system administrators tune their system in order to achieve better energy efficiency.

Acknowledgment

This work is part of the BigStorage project, funded by the European Union under the Marie Sklodowska-Curie Actions (H2020-MSCA-ITN-2014-642963).

6. REFERENCES

[1] Energy consumption in US Datacenters. http://www.nrdc.org/energy/data-center-efficiency-assessment.asp, 2016. [Online; accessed September-2016].

[2] GRID'5000. www.grid5000.fr/, 2016. [Online; accessed September-2016].

[3] Riak. www.basho.com/products/riak-kv/, 2016. [Online; accessed September-2016].

[4] H. Amur, J. Cipar, V. Gupta, G. R. Ganger, M. A. Kozuch, and K. Schwan. Robust and flexible power-proportional storage. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, pages 217–228, New York, NY, USA, 2010. ACM.

[5] B. Atikoglu, Y. Xu, E. Frachtenberg, S. Jiang, and M. Paleczny. Workload analysis of a large-scale key-value store. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '12, pages 53–64, 2012.

[6] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears. Benchmarking cloud serving systems with ycsb. In *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, pages 143–154, 2010.

[7] A. Dragojević, D. Narayanan, M. Castro, and O. Hodson. Farm: Fast remote memory. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 401–414, Seattle, WA, Apr. 2014.

[8] B. Fan, D. G. Andersen, and M. Kaminsky. Memc3: Compact and concurrent memcache with dumber caching and smarter hashing. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 371–384, Lombard, IL, 2013.

[9] M. Iravani. How twitter uses redis to scale - 105tb ram, 39mm qps, 10,000+ instancess, 2015.

[10] C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T. W. Keller. Energy management for commercial servers. *Computer*, 36(12):39–48, Dec 2003.

[11] H. Li, A. Ghodsi, M. Zaharia, S. Shenker, and I. Stoica. Tachyon: Reliable, memory speed storage for cluster computing frameworks. In *Proceedings of the ACM Symposium on Cloud Computing*, SOCC '14, pages 6:1–6:15, 2014.

[12] H. Lim, D. Han, D. G. Andersen, and M. Kaminsky. Mica: A holistic approach to fast in-memory key-value storage. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 429–444, Seattle, WA, Apr. 2014.

[13] C. Mitchell, Y. Geng, and J. Li. Using one-sided rdma reads to build a fast, cpu-efficient key-value store. In *Presented as part of the 2013 USENIX Annual Technical Conference (USENIX ATC 13)*, pages 103–114, San Jose, CA, 2013.

[14] R. Nishtala, H. Fugal, S. Grimm, M. Kwiatkowski, H. Lee, H. C. Li, R. McElroy, M. Paleczny, D. Peek, P. Saab, D. Stafford, T. Tung, and V. Venkataramani. Scaling memcache at facebook. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 385–398, Lombard, IL, 2013.

[15] D. Ongaro, S. M. Rumble, R. Stutsman, J. Ousterhout, and M. Rosenblum. Fast crash recovery in ramcloud. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 29–41, 2011.

[16] A. Papagiannis, G. Saloustros, P. González-Férez, and A. Bilas. Tucana: Design and implementation of a fast and efficient scale-up key-value store. In *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, pages 537–550, Denver, CO, June 2016.

[17] T. Rabl, S. Gómez-Villamor, M. Sadoghi, V. Muntés-Mulero, H.-A. Jacobsen, and S. Mankovskii. Solving big data challenges for enterprise application performance management. *Proc. VLDB Endow.*, 5(12):1724–1735, Aug. 2012.

[18] S. M. Rumble, A. Kejriwal, and J. Ousterhout. Log-structured memory for dram-based storage. In *Proceedings of the 12th USENIX Conference on File and Storage Technologies (FAST 14)*, pages 1–16, Santa Clara, CA, 2014.

[19] E. Thereska, A. Donnelly, and D. Narayanan. Sierra: Practical power-proportionality for data center storage. In *Proceedings of the Sixth Conference on Computer Systems*, EuroSys '11, pages 169–182, New York, NY, USA, 2011. ACM.

[20] Y. Wang, L. Zhang, J. Tan, M. Li, Y. Gao, X. Guerin, X. Meng, and S. Meng. Hydradb: A resilient rdma-driven key-value middleware for in-memory cluster computing. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '15, pages 22:1–22:11, 2015.