An In-Depth Analysis of Cloud Block Storage Workloads in Large-Scale Production

Jinhong Li[†], Qiuping Wang[†], Patrick P. C. Lee[†], and Chao Shi[*]
[†]*The Chinese University of Hong Kong* [*]*Alibaba Group*

***Abstract*—Cloud block storage systems support diverse types of applications in modern cloud services. Characterizing their I/O activities is critical for guiding better system designs and optimizations. In this paper, we present an in-depth analysis of production cloud block storage workloads through the block-level I/O traces of billions of I/O requests collected from Alibaba Cloud. We study the characteristics of load intensity, spatial patterns, and temporal patterns. Also, we present a comparative study on our traces and the notable public block-level I/O traces from Microsoft Research Cambridge, and identify the commonalities and differences of the two sets of traces. Finally, we provide 15 findings and discuss their implications on load balancing, cache efficiency, and storage cluster management in a cloud block storage system. Our traces are now released for public use.**

I. Introduction

Traditional desktop and server applications, such as virtual desktops, operating systems, web services, relational databases, and key-value stores, are now moving to the cloud. *Cloud block storage* systems [1], [2], [13], [17], [18], [34] form an infrastructure that allows cloud service providers to manage large-scale physical storage clusters. They also provide virtual disks, called *volumes*, for clients to host various types of applications. To allow performance optimizations and efficient resource provisioning of cloud block storage systems, it is critical to characterize and understand the I/O behaviors of the applications in production environments.

Several field studies have analyzed the I/O behaviors of various architectures via the collection and characterization of block-level I/O traces [3], [9], [11], [12], [21], [35]. In particular, the public block-level I/O traces released by Microsoft Research Cambridge [21] have received wide attention from researchers and practitioners. The traces, which we refer to as *MSRC*, have been extensively analyzed to motivate storage system designs and optimizations, such as I/O scheduling [6], [14], [15], [21], caching [23], [24], erasure-coded storage [7], [34], as well as cloud block storage [13].

However, the MSRC traces, which were collected from enterprise data centers more than a decade ago, may not necessarily reflect the actual I/O behaviors of today's cloud block storage systems. Modern cloud environments often host much more diverse types of applications, some of which feature unique characteristics (e.g., short-lived tasks [20]) that are not commonly found in traditional data center environments. Also, the overall workloads in MSRC are read-dominant [21], while the workloads in cloud environments are often write-dominant due to the heavy use of read caches in cloud applications [16], [30]. Such deficiencies motivate the need of collecting and analyzing comprehensive block-level I/O traces from real-world cloud block storage systems in large-scale production.

In this paper, we present an in-depth study on the block-level I/O traces collected from a production cloud block storage system deployed at Alibaba Cloud. Our traces, which we refer to as *AliCloud*, cover one-month I/O activities of 1,000 volumes, totaling billions of I/O requests and hundreds of terabytes of I/O traffic. We characterize the I/O behaviors in terms of the load intensity, spatial patterns, and temporal patterns. In particular, we present a comparative analysis on both the AliCloud and MSRC traces, and identify the commonalities and differences of the two traces. To this end, we highlight 15 findings, and provide insights into load balancing, cache efficiency, and storage cluster management in cloud block storage systems. To the best of our knowledge, our trace analysis is one of the largest field studies on block-level I/O traces reported in the literature [3], [9], [11], [12], [21], [35].

We highlight some major findings of our trace analysis. For load intensity, both traces show similar amounts of I/O traffic, while AliCloud shows more diverse workloads. For spatial patterns, both traces show aggregations of reads and writes in small working sets, while the aggregations of reads and writes in AliCloud are more prominent. Also, both traces show varying patterns in update coverage. For temporal patterns, both traces have varying temporal update patterns across volumes. However, they show different access tendencies for the written blocks: each written block in AliCloud is likely followed by a write, while that in MSRC is about equally likely to be followed by either a read or a write.

We now release the block-level I/O traces of AliCloud at: **https://github.com/alibaba/block-traces**. We hope that the community can use the traces to not only validate our analysis, but also drive new designs and optimizations for cloud block storage and generally large-scale storage systems.

II. Background and Methodology

We introduce the cloud block storage architecture considered in the paper (Section II-A). We further elaborate how our trace analysis should characterize the I/O activities in response to the design considerations for cloud block storage (Section II-B).

A. Cloud Block Storage

Figure 1 depicts the architecture of a cloud block storage system considered in the paper. The cloud block storage system serves as a middleware layer that bridges: (i) the virtual

978-1-7281-7645-1/20/$31.00 ©2020 IEEE
DOI 10.1109/IISWC50251.2020.00013

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.

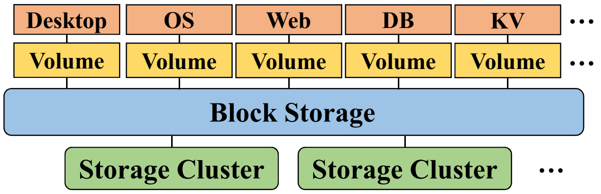


Fig. 1: Architecture of a cloud block storage system. It comprises multiple volumes that host a mix of cloud applications (e.g., virtual desktops, operating systems, web services, relational databases, and key-value stores).

disks (referred to as the *volumes*) that are perceived by upper-layer applications, and (ii) the storage clusters that provide physical storage space owned by cloud service providers. Each application is allocated with a dedicated volume. It issues read or write requests through the dedicated volume to the storage clusters. Each volume is typically replicated across multiple storage clusters for fault tolerance. For performance and reliability, today's storage cluster are often backed by flash-based solid-state drives (SSDs) instead of hard disk drives [32].

In production, a cloud block storage system may manage diverse types of upper-layer cloud applications (Figure 1). The I/O characteristics of such applications are often largely different, as we show in this paper.

B. Analysis Methodology

Cloud block storage systems should maintain quality-of-services guarantees (e.g., low-latency requests and fairness) and efficient resource utilizations (e.g., long device lifetime). We highlight three design considerations of cloud block storage systems, namely *load balancing*, *cache efficiency*, and *storage cluster management*. In the following, we explain how each of the design considerations can be addressed in our trace analysis of I/O activities in cloud block storage.

Load balancing. Maintaining load balancing across storage devices is important for availability. If load imbalance exists, some storage devices may be overloaded by a large number of I/O requests and cannot serve incoming requests in a timely manner, thereby increasing the overall I/O latencies. In addition, the overloading of I/O requests may aggravate flash wearing [32], leading to reduced endurance. Since load balancing addresses the performance differences due to the uneven distribution of I/O traffic, our trace analysis should examine the load intensities of I/O traffic.

Cache efficiency. To speed up I/O performance, storage systems typically cache frequently accessed data based on efficient resource allocation schemes and admission policies [4], [29]. However, the high variations of I/O characteristics may introduce improper cache space allocation and cache management policies, which degrade hit ratios and increase the overall I/O latencies. To investigate how the caching design can leverage workload characteristics, our trace analysis should address the spatial and temporal aggregations of I/O traffic.

Storage cluster management. Enterprise storage clusters increasingly move to flash-based storage, which is sensitive to varying workload patterns in both performance and endurance. In particular, the update patterns can determine the effectiveness of garbage collection and wear-leveling in flash [10]. Storage cluster management should address the variations of workload patterns, so as to maintain high performance and endurance of the underlying flash devices. Thus, our trace analysis should focus on the spatial and temporal patterns for update requests. Also, as small and random I/Os can degrade the performance and endurance of flash storage [19], our trace analysis should also examine the randomness of I/Os.

III. Traces

We describe two sets of traces used in our analysis (Section III-A) and state the limitations of our trace analysis (Section III-B). We present a high-level analysis on the basic statistics as well as the commonalities and differences on both traces (Section III-C).

A. Trace Overview

Our trace analysis is based on two sets of block-level I/O traces collected from different production environments. For brevity, we refer to the traces as *AliCloud* and *MSRC* in short in the following discussion.

AliCloud. The traces were collected from a cloud block storage system deployed at Alibaba Cloud over a one-month period in January 2020. They comprise block-level I/O requests collected from 1,000 volumes, each of which has a raw capacity from 40 GiB to 5 TiB. The workloads span diverse types of cloud applications (Section II-A). Each collected I/O request specifies the volume number, request type, request offset, request size, and timestamp.

MSRC [21]. The traces were collected by Microsoft Research Cambridge from a data center of Microsoft Windows servers over a one-week period in February 2007. They comprise one-week block-level I/O requests from 36 volumes over 179 disks on 13 servers. The workloads span a variety of applications, including home directories, project directories, web services, source control, media services, etc. Each collected I/O request includes all the fields as in AliCloud; in addition, it also includes the response time of the request.

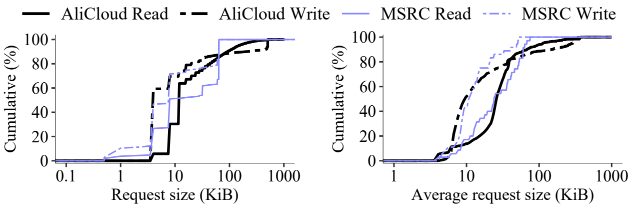
B. Limitations

Our trace analysis has several limitations due to the unavailable information in AliCloud. First, the traces do not record the response times of the I/O requests as in MSRC, so we cannot conduct latency analysis on I/O requests in actual deployment. Also, the traces do not indicate the specific applications running atop individual volumes, so we cannot investigate the relationship between specific application workloads and their I/O patterns. Finally, the traces do not capture the information of physical storage devices (e.g., data placement and failure statistics), so we cannot study the performance and reliability correlations between the cloud block storage system and the underlying storage clusters.

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.

	AliCloud	MSRC
#Volumes	1,000	36
Duration (days)	31	7
#Reads (millions)	5,058.6	304.9
#Writes (millions)	15,174.4	128.9
Read Traffic (TiB)	161.6	9.04
Write Traffic (TiB)	455.5	2.39
Update Traffic (TiB)	429.2	2.01
Total WSS (TiB)	29.5	2.87
Read WSS (TiB)	10.1	2.82
Write WSS (TiB)	26.3	0.38
Update WSS (TiB)	18.6	0.17

TABLE I: Basic statistics of both AliCloud and MSRC.



(a) Cumulative distribution of request sizes across all requests

(b) Cumulative distribution of average request sizes across all volumes

Fig. 2: Cumulative distributions of I/O request sizes.

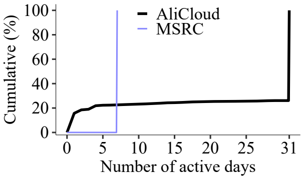


Fig. 3: Cumulative distributions of numbers of active days across all volumes.

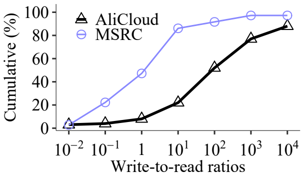


Fig. 4: Cumulative distributions of write-to-read ratios across all volumes.

C. High-level Analysis

We now present a high-level analysis on both AliCloud and MSRC by collectively analyzing the I/O requests of all volumes in each of the traces and presenting the overall basic statistics. We further identify some of their commonalities and differences. Table I shows different categories of basic statistics of both AliCloud and MSRC, including (i) the numbers of reads and writes, (ii) the total amounts of data read, written, and updated, as well as (iii) the working set sizes (WSSs) of reads, writes, and updates.

AliCloud has a much larger scale than MSRC. Referring to Table I, AliCloud has a much larger scale than MSRC in various aspects, including the number of volumes, the trace duration, the number of I/O requests, and the size of I/O traffic; to our knowledge, the scale of AliCloud is among the largest block-level I/O traces reported in the literature. Specifically, AliCloud contains 20.2 billion I/O requests, 46.6× the total number of I/O requests in MSRC. It also has much larger scales, in terms of the number of volumes (27.8×), the I/O traffic size (54.1×), and the WSS (10.3×), compared to those in MSRC.

Reads span a small proportion of working sets in AliCloud. Referring to Table I, reads in AliCloud only occupy 34.3% of the total WSS, while reads in MSRC occupy a much larger proportion (98.4%) of the total WSS. On the other hand, writes in AliCloud occupy 89.4% of the total WSS. The results indicate that a substantial amount of written data is never read again. One possible reason is that a large fraction of applications (e.g., backups or journaling) tend to only write data but rarely read data.

Small-size I/Os dominate in both AliCloud and MSRC. Figure 2(a) shows the cumulative distributions of request sizes across all I/O requests in both AliCloud and MSRC. We see that both traces feature small-size I/O requests (less than 100 KiB). Specifically, in AliCloud, 75% of reads and writes are no larger than 32 KiB and 16 KiB, respectively, while in MSRC, 75% of reads and writes are no larger than 64 KiB and 20 KiB, respectively.

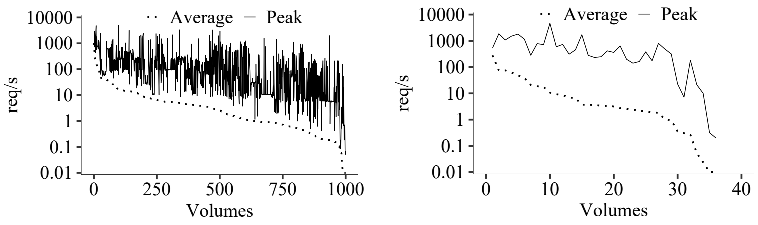
The dominance of small-size I/O requests also holds in individual volumes. We compute the average request size for each volume. Figure 2(b) shows the cumulative distributions of the average request sizes of all volumes in both AliCloud and MSRC. We see that 75% of the average read and write sizes in AliCloud are less than 39.1 KiB and 34.4 KiB, respectively, while 75% of the average read and write sizes in MSRC are less than 50.8 KiB and 15.3 KiB, respectively.

A non-negligible fraction of volumes in AliCloud are active in short time periods. We study the activeness of individual volumes. Here, we measure the number of active days for each volume, in which a volume is said to be active if it receives at least one I/O request (i.e., up to 31 and 7 active days in AliCloud and MSRC, respectively). Figure 3 depicts the cumulative distributions of numbers of active days across all volumes in both AliCloud and MSRC. In AliCloud, 15.7% of volumes are active for only one day, while all volumes in MSRC are active for all 7 days in the entire trace duration. One possible reason for the short active periods in such volumes in AliCloud is the presence of short-lived tasks in cloud applications [20].

Most volumes in AliCloud are write-dominant. Referring to Table I, the overall write-to-read ratio (i.e., the ratio between the number of writes and the number of reads) in AliCloud is 3:1, while that in MSRC is 0.42:1. We further analyze the write-to-read ratios on a per-volume basis. Figure 4 shows the cumulative distributions of write-to-read ratios across all volumes in both AliCloud and MSRC. In AliCloud, 91.5% (915 out of 1,000) of volumes are write-dominant (i.e., the write-to-read ratios are larger than 1). Also, almost half (42.4%) of the volumes even have very high write-to-read ratios that are larger than 100. This is in contrast to MSRC, in which only 53% (19 out of 36) of volumes are write-dominant. A possible reason is the wide use of application-level read caches, which absorb reads in the application layer without issuing them to the storage layer [30].

Summary. Both AliCloud and MSRC have some common

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.



(a) AliCloud (b) MSRC

Fig. 5: Finding 1. Average and peak intensities of volumes (sorted by average intensities in descending order).

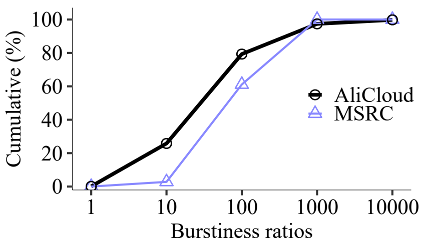


Fig. 6: Findings 2-3. Cumulative distribution of burstiness ratios of volumes.

aspects, such as the dominance of small-size I/O requests, yet they also have many differences. In particular, AliCloud has the following unique aspects: a very large scale, a small WSS for reads, a non-negligible fraction of volumes with short active periods, and high write-to-read ratios in most volumes.

IV. Findings

In this section, we conduct an in-depth analysis on both AliCloud and MSRC in three aspects: load intensity, spatial patterns, and temporal patterns. We report 15 findings from our analysis.

A. Load Intensity

We study the load intensity characteristics of volumes in both AliCloud and MSRC through a number of metrics. Specifically, we examine the average and peak load intensities [21] and the distribution of inter-arrival times of requests [27]. We also examine the activeness of volumes through the number of active volumes [21] and the active period of each volume.

Finding 1. *Both AliCloud and MSRC have similar load intensities of volumes.*

We measure the load intensities of individual volumes, in terms of the number of requests per second (req/s), in two aspects. We first measure the *average intensity* of a volume, defined as the total number of requests divided by the time elapsed between the first and last requests of the volume. We also measure the *peak intensity* of a volume, in which we divide the whole duration of requests of the volume into one-minute intervals and find the peak intensity as the maximum number of requests (per minute) across all intervals.

Figure 5 shows the average and peak intensities of volumes in both AliCloud and MSRC, sorted by the average intensities of volumes in descending order. We observe similar trends of average and peak intensities in both traces. In AliCloud and MSRC, only 1.90% and 2.78% of volumes have average intensities above 100 req/s, while 81.6% and 72.2% of volumes have average intensities lower than 10 req/s, respectively. Their medians of average intensities are 2.55 req/s and 3.36 req/s, respectively. Furthermore, the maximum peak intensities in AliCloud and MSRC are 4,926.8 req/s and 4,633.6 req/s, respectively.

Finding 2. *Both AliCloud and MSRC have high burstiness in a non-negligible fraction of volumes, but have overall low burstiness.*

Traces	AliCloud	MSRC
Peak intensity (req/s)	15,965.8	5,296.8
Average intensity (req/s)	7,554.1	717.0
Burstiness ratio	2.11	7.39

TABLE II: Finding 2: Overall peak and average intensities as well as burstiness ratios.

We examine the burstiness of both traces. We measure the *burstiness ratio* of a volume, defined as the ratio between the peak intensity and the average intensity of the volume. Figure 6 shows the cumulative distributions of burstiness ratios across all volumes in both AliCloud and MSRC. We see that a substantial fraction of volumes (20.7% in AliCloud and 38.9% in MSRC) have burstiness ratios higher than 100. This implies that such volumes can observe load imbalance at some time. On the other hand, if we examine overall burstiness level by aggregating all volumes of the whole traces, the burstiness ratios are mild, with 2.11 for AliCloud and 7.39 for MSRC (see Table II).

Finding 3. *AliCloud has more diverse burstiness across volumes than MSRC.*

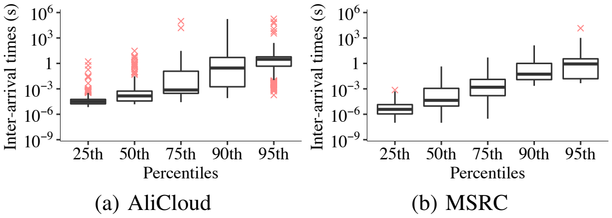
The volumes in AliCloud span a wider range of burstiness than those in MSRC. Referring to Figure 6, for the volumes with low burstiness, 25.8% of volumes in AliCloud have burstiness ratios less than 10, while in MSRC, the corresponding percentage of volumes is only 2.78%. On the other hand, for the volumes with high burstiness, 2.60% of volumes in AliCloud have burstiness ratios larger than 1,000, while there is no such volume in MSRC. The high diversity of burstiness in AliCloud suggests large variations in workload characteristics.

Finding 4. *Both AliCloud and MSRC have high short-term burstiness from the perspective of inter-arrival times of requests.*

We measure the inter-arrival times of I/O requests (i.e., the elapsed time between two adjacent requests) for each volume. Also, we consider five groups of percentiles of inter-arrival times for each volume, including the 25th, 50th, 75th, 90th, and 95th percentiles. We represent each group of percentile values of all volumes by boxplots.

Figure 7 shows the results of both AliCloud and MSRC. Both traces have a high number of bursty requests, as indicated by large fractions of short inter-arrival times in the volumes. In particular, the medians of the groups of 25th, 50th, and 75th percentiles are lower than 1.3 ms, or equivalently over

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.



(a) AliCloud

(b) MSRC

Fig. 7: Finding 4. Inter-arrival times of requests. Each boxplot represents the distribution of all the values collected in each volume according to the corresponding percentile.

700 req/s (i.e., 31 μs, 145 μs, and 735 μs in AliCloud, and 3.5 μs, 30.5 μs, and 1.3 ms in MSRC, respectively).

Also, the volumes in AliCloud have much higher inter-arrival times of requests than those in MSRC. For example, half of the volumes in AliCloud has 25th percentiles higher than 31 μs (Figure 7(a)), while half of the volumes in MSRC have 25th percentiles higher than 3.5 μs (Figure 7(b)). The same observation holds for other groups of percentiles.

Finding 5. *Most of the volumes in both AliCloud and MSRC are active throughout the trace periods, while AliCloud has higher activeness than MSRC.*

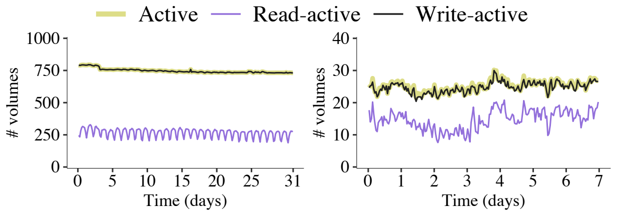
Recall from Section III-C that we examine the activeness of volumes of both AliCloud and MSRC on a per-day basis. We now revisit the activeness of volumes of both traces in a more fine-grained manner. Specifically, we divide the traces into 10-minute intervals. We say that a volume is *active* in an interval if it has at least one request in the interval. We also say that a volume is *read-active* and *write-active* in an interval if it has at least one read request and one write request in the interval, respectively.

Figure 8 depicts the numbers of active, read-active, and write-active volumes throughout the trace periods in both AliCloud and MSRC. More than 59.4% of volumes in both traces are active throughout the whole trace periods. Also, the number of active volumes in AliCloud has a more stable trend compared to that in MSRC.

We also measure the active time period of each volume, based on the number of 10-minute intervals in which the volume is active. Figure 9 depicts the cumulative percentages of active time periods across all volumes in both AliCloud and MSRC. More than 72.2% and 55.6% of the volumes are active during 95% of the whole trace periods in AliCloud and MSRC, respectively. This indicates that most of the volumes in both AliCloud and MSRC have high activeness throughout the whole trace periods, and AliCloud has higher activeness in general than MSRC.

Finding 6. *Writes are the dominant factor in determining activeness in both AliCloud and MSRC.*

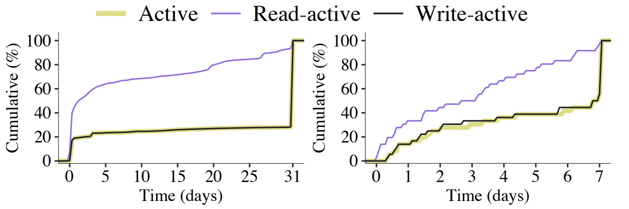
Referring to both Figures 8 and 9, the curves of "Active" and "Write-active" nearly overlap with each other in both AliCloud and MSRC. It suggests that the activeness of both traces (in terms of the number of active volumes and the active time period of a volume) is mainly determined by the presence of writes.



(a) AliCloud

(b) MSRC

Fig. 8: Findings 5-7. Numbers of active, read-active, and write-active volumes. Note that the "Active" and "Write-active" curves almost overlap with each other.



(a) AliCloud

(b) MSRC

Fig. 9: Findings 5-7. Cumulative distributions of active time periods across all volumes. Note that the "Active" and "Write-active" curves almost overlap with each other.

Finding 7. *Removing write requests shows drastic decreases in activeness in both AliCloud and MSRC. AliCloud is also less read-active than MSRC.*

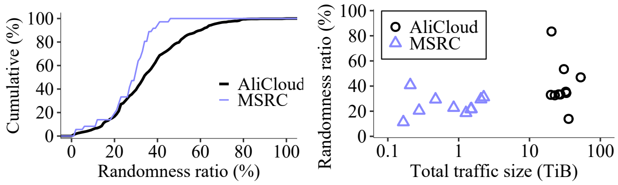
If we remove write requests and consider only the read-active volumes, Figure 8 shows that the number of active volumes decreases drastically. In particular, the number of active volumes reduces by 58.3-73.6% in AliCloud (Figure 8(a)), while the reduction is 24.6-65.8% in MSRC (Figure 8(b)).

Figure 9 shows that removing write requests also causes the volumes in both AliCloud and MSRC to have low read-active time. In AliCloud, half of the volumes have less than only 1.28 days of read-active time after removing writes, and only 7.9% of the volumes can reach more than 30 days of read-active time (Figure 9(a)). In MSRC, half of the volumes are read-active for less than 2.66 days, and only 16.7% of the volumes can reach more than 6 days of read-active time (Figure 9(b)). This suggests that removing writes produces a high level of idle periods, such that we can apply write offloading to cloud block storage for power savings [21].

B. Spatial Patterns

We study the spatial characteristics of volumes in both AliCloud and MSRC through the following metrics. First, we study the randomness of I/O requests by examining the offset differences of recent requests [3], [25], as random I/Os can compromise the performance and endurance of flash-based storage [19]. Second, we examine the aggregations of reads and writes in working sets, so as to provide hints for resource allocation in caching [13], [14], [24]. Finally, we examine the patterns of update coverage (i.e., the percentage of WSS for

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.



(a) Randomness ratio distribution (b) Randomness ratio vs. traffic

Fig. 10: Finding 8. Cumulative distributions of randomness ratios of volumes (figure (a)) and the relationship between the randomness ratios and total traffic sizes in top-10 traffic-intensive volumes.

updates), which is important for optimizing update performance in the storage cluster management [7].

Finding 8. *Random I/Os are common in both AliCloud and MSRC. The volumes in AliCloud see more random I/Os than those in MSRC.*

We study the randomness of I/O requests by examining the spatial relationships among adjacent requests. To quantify the randomness of a request, we measure the minimum distance between the current offset of the request and the offsets of the previous 32 requests [3], [25]. If the minimum distance exceeds a threshold (e.g., 128 KiB [25]), we regard the request as a random request. We measure the *randomness ratio* of a volume, defined as the percentage of random requests over all requests.

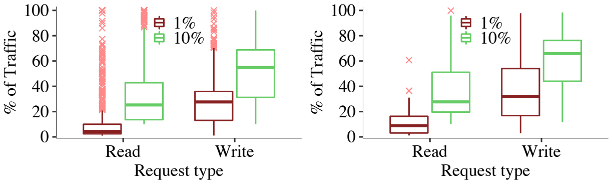
Figure 10(a) shows the cumulative distributions of randomness ratios of volumes in both AliCloud and MSRC. AliCloud in general shows a higher randomness ratio than MSRC. In particular, all volumes in MSRC have less than 46% of random requests, while 20% of volumes in AliCloud have more than 50% of random requests.

We further examine the randomness ratios of the top-10 volumes that have the most I/O traffic in each trace. Figure 10(b) shows the relationships between the randomness ratios and the total I/O traffic sizes of the top-10 volumes. We see that the volumes with large amounts of I/O traffic have high randomness ratios in general. The randomness ratios of the top-10 volumes in AliCloud and MSRC are 13.9-83.4% and 11.3-40.8%, respectively, and their I/O traffic sizes are 20.0-52.8 TiB and 0.17-2.26 TiB, respectively. The results indicate that random I/Os are also common in traffic-intensive volumes.

Combining with the observation that small-size I/O requests dominate in both traces (Section III-C), we see that random and small I/Os are common in both traces (especially in AliCloud). Such access patterns can compromise the performance and endurance of flash-based storage [19].

Finding 9. *Reads and writes aggregate in small working sets for a non-negligible fraction of volumes in both AliCloud and MSRC. Writes are more aggregated than reads.*

We study how reads and writes aggregate in the working sets for each volume. Specifically, in the read (or write) working sets, we focus on the top-1% and top-10% of unique blocks that receives most read (or write) traffic. We examine the percentages of read (or write) traffic size of such blocks over the total read (or write) traffic size; a higher percentage implies that the I/O traffic is more aggregated in such blocks.



(a) AliCloud (b) MSRC

Fig. 11: Finding 9. Boxplots of percentages of traffic sizes for the top-1% and top-10% read and write blocks across all volumes.

Figure 11 shows the boxplots of percentages of traffic sizes for the top-1% and top-10% blocks across all volumes in AliCloud and MSRC. We see that the read and write traffic can aggregate in the top-1% and top-10% blocks in a non-negligible fraction of volumes. In AliCloud, 75% of volumes have at least 2.5% and 13.6% of read traffic in the top-1% and top-10% read blocks, respectively (Figure 11(a)), while in MSRC, the corresponding percentages of read traffic are 3.1% and 19.6%, respectively (Figure 11(b)).

In AliCloud, the boxplots show 147 volumes as outliers in top-1% read blocks (Figure 11(a)). Such outlier volumes have more than 21.3% of read traffic in their top-1% read blocks. It implies that a small read cache can absorb a substantial amount of read traffic for such volumes.

Compared to reads, writes are more aggregated as in the figure. In AliCloud, the 25th percentiles of read traffic in top-1% and top-10% read blocks are 2.5% and 13.6% respectively, while the 25th percentiles of write traffic in top-1% and top-10% write blocks increase to 13.0% and 31.2%, respectively (Figure 11(a)). Similar observations hold in MSRC (Figure 11(b)).

Finding 10. *Reads and writes tend to aggregate in read-mostly and write-mostly blocks in AliCloud, respectively.*

We further classify the blocks into different types as in [14] and examine the aggregation of reads and writes. Specifically, we classify a block as *read-mostly* (or *write-mostly*) if its read (or write) traffic occupies more than 95% of its total I/O traffic. We examine the percentage of read (or write) traffic that goes to read-mostly (or write-mostly) blocks.

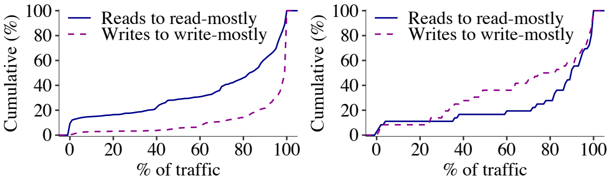
Table III shows the overall percentages of read and write traffic that goes to read-mostly and write-mostly blocks in both AliCloud and MSRC. In AliCloud, the majority of read traffic (59.2%) and write traffic (80.7%) goes to read-mostly blocks and write-mostly blocks, respectively. In MSRC, 75.9% of read traffic goes to read-mostly blocks; however, only 33.5% of write traffic goes to write-mostly blocks. Note that the limited aggregation of writes in write-mostly blocks in MSRC is inconsistent with the prior finding in [14]. The reason is that the study in [14] considers only 12 out of 36 volumes in MSRC, while we consider all 36 volumes.

Figure 12 shows the cumulative distributions of percentages of read and write traffic that goes to read-mostly and write-mostly blocks, respectively, across all volumes in both AliCloud

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.

Traces	AliCloud	MSRC
Reads to read-mostly blocks (%)	59.2	75.9
Writes to write-mostly blocks (%)	80.7	33.5

TABLE III: Finding 10. Percentages of read and write traffic going to read-mostly and write-mostly blocks, respectively.



(a) AliCloud

(b) MSRC

Fig. 12: Finding 10. Cumulative distributions of percentages of read and write traffic going to read-mostly and write-mostly blocks, respectively, across all volumes.

Traces	AliCloud	MSRC
Mean (%)	76.6	36.2
Median (%)	61.2	9.4
90th percentiles (%)	92.1	63.0

TABLE IV: Finding 11. Means, medians, and 90th percentiles of update coverage of all volumes.

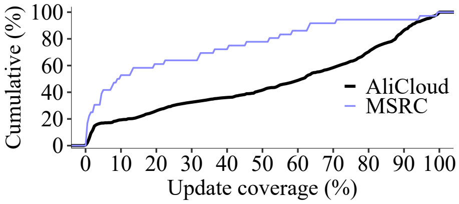


Fig. 13: Finding 11. Cumulative distributions of update coverage percentages across all volumes.

and MSRC. Most of the volumes in both traces have high percentages of read and write traffic aggregated in read-mostly and write-mostly blocks, respectively. In AliCloud, half of the volumes have more than 83% of reads going to read-mostly blocks and more than 99% writes going to write-mostly blocks (Figure 12(a)). In MSRC, the corresponding percentages are 90% and 75%, respectively (Figure 12(b)).

Finding 11. *AliCloud generally has higher update coverage than MSRC. The update coverage also varies across volumes.*

Recall that Table I (Section III-C) shows the overall WSSs (working set sizes) for reads, writes, and updates. We now examine the spatial characteristics of updates. We focus on the update working set, which covers the blocks that are written more than once. We measure the *update coverage* of a volume, defined as the ratio between the update WSS and the total WSS of the volume [7].

Table IV shows the averages, medians, and 90th percentiles of update coverage of all volumes in both AliCloud and MSRC. In general, AliCloud has higher update coverage than MSRC. In AliCloud, half of volumes have larger than 61.2% of update coverage, while in MSRC, the corresponding percentage is 9.4% only. This suggests that AliCloud is more update-intensive than MSRC.

Figure 13 shows the cumulative distributions of update coverage percentages across all volumes in both AliCloud and MSRC. In AliCloud, the update coverage is diverse, in which 45.2% of volumes have update coverage larger than 65%. On the other hand, in MSRC, 33 out of 36 volumes have update coverage below 65%.

C. Temporal Patterns

We study the temporal characteristics of volumes in both AliCloud and MSRC by examining the temporal relationship of adjacent I/O requests. We first examine the time elapsed between adjacent requests to the same block with respect to different combinations of read and write requests for workload-aware caching designs [24]. We also study the update interval (i.e., the time interval between two consecutive writes to the same block), which facilitates flash-based storage management. [6], [15]. Finally, we study the miss ratios under least recently used (LRU) caching, which reflects the temporal aggregation of traffic for caching efficiency [28], [31].

Finding 12. *Both AliCloud and MSRC have large read-after-write (RAW) time, but small write-after-write (WAW) time. In AliCloud, the number of WAW requests is larger than that of RAW requests.*

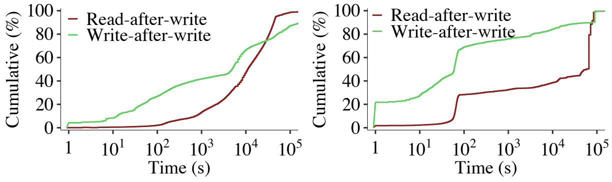
We first examine two types of adjacent requests [24]: (i) a *read-after-write (RAW)* request, which refers to the read following immediately the write to the same block; and (ii) a *write-after-write (WAW)* request, which refers to the write following immediately the write to the same block. We measure the time of a RAW (resp. WAW) request as the elapsed time between the adjacent read and write (resp. the two adjacent writes) to the same block.

Figure 14 shows the cumulative distributions of RAW and WAW times across all RAW and WAW requests, respectively, in both AliCloud and MSRC. Both AliCloud and MSRC have large RAW time. Specifically, the 50th percentiles of the RAW time in AliCloud and MSRC are 3.0 hours and 16.2 hours, respectively. Also, the numbers of RAW requests that exceed 5 minutes are 93.3% and 68.8% in AliCloud and MSRC, respectively; such findings are consistent with those in the prior work [24].

On the other hand, both AliCloud and MSRC have small elapsed time in WAW requests, as shown in Figure 14. In particular, the 50th percentiles of the WAW time in AliCloud and MSRC are 1.4 hours and 0.2 hours, respectively. Also, 22.4% and 50.6% of WAW times in AliCloud and MSRC are less than 1 minute, respectively. This suggests that write caching can effectively absorb the subsequent writes to the same block and hence reduce the load for primary storage.

Table V shows the numbers of RAW and WAW requests in both AliCloud and MSRC. We observe a large difference in the numbers of RAW and WAW requests in AliCloud, but a small difference in MSRC. Specifically, in AliCloud, the numbers of RAW and WAW requests are 12.4 billion and 103.7 billion,

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.



(a) AliCloud

(b) MSRC

Fig. 14: Finding 12. Cumulative distributions of RAW and WAW times across all RAW and WAW requests, respectively.

Traces	RAW (M)	WAW (M)	RAR (M)	WAR (M)
AliCloud	12,432.7	103,708.4	29,845.0	11,760.6
MSRC	297.2	289.8	1,382.6	330.0

TABLE V: Findings 12-13. Numbers of RAW, WAW, RAR, and WAR requests (in millions).

respectively; the number of WAW requests is 8.4× that of RAW requests. In MSRC, those numbers are 297.2 million and 289.8 million, respectively, and are close to each other.

Finding 13. *Most of the read-after-read (RAR) and write-after-read (WAR) requests in AliCloud have large elapsed time, while the RAR and WAR requests in MSRC generally have very small elapsed time. In both traces, the WAR time is much larger than the RAR time, while the numbers of RAR and WAR requests are comparable.*

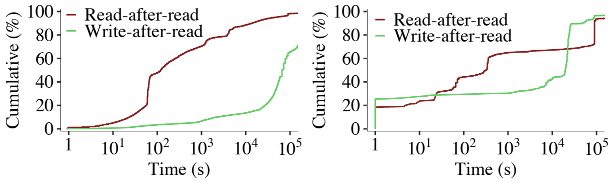
We further examine two types of adjacent requests: (i) a *read-after-read (RAR)* request, which refers to the read following immediately the read to the same block; and (ii) a *write-after-read (WAR)* request, which refers to the write following immediately the read to the same block.

Figure 15 shows the cumulative distributions of RAR and WAR times across all RAR and WAR requests, respectively, in both AliCloud and MSRC. In AliCloud, most of the RAR and WAR times are larger than 1 minute, while only 22.1% and 2.8% of RAR and WAR times are less than 1 minute, respectively (Figure 15(a)). However, in MSRC, there exist non-negligible fractions of RAR and WAR times (35.6% and 29.2%, respectively) that are less than 1 minute. Also, at least 18.5% and 25.4% of RAR and WAR times are smaller than 1 second, respectively (Figure 15(b)).

Overall, in both AliCloud and MSRC, the WAR time is generally larger than the RAR time. In AliCloud, the 50th percentiles of RAR and WAR times are 2.0 minutes and 18.3 hours, respectively, while 21.0% and 88.8% of RAR and WAR times are larger than 1 hour, respectively (Figure 15(a)). In MSRC, the 50th percentiles of RAR and WAR times are 5.0 minutes and 5.5 hours, respectively, while 33.6% and 66.7% of RAR and WAR times are larger than 1 hour, respectively. The results indicate that a block being read is likely read again soon.

We also examine the numbers of RAR and WAR requests in both AliCloud and MSRC, as shown in Table V. In AliCloud and MSRC, the numbers of RAR requests are 2.54× and 4.19× those of WAR requests, respectively.

Finding 14. *Written blocks have varying update intervals.*



(a) AliCloud

(b) MSRC

Fig. 15: Finding 13. Cumulative distributions of RAR and WAR times across all RAR and WAR requests, respectively.

We measure the *update interval* of a block, defined as the elapsed time between two consecutive writes to the same block. Note that the update interval differs from the WAW time, as the former allows reads between two writes. Each block may be written more than once, so it may be associated with multiple update intervals (e.g., a block that is written M times has $M-1$ update intervals). The update interval of a block describes the lifetime of the block data.

Table VI shows the update intervals in different groups of percentiles across all volumes in AliCloud and MSRC. In AliCloud, the update intervals generally have long durations, while in MSRC, the update intervals generally have short durations. In AliCloud, 50% of update intervals are larger than 1.6 hours, and the 90th percentile is 50.3 hours. In MSRC, the update intervals have a bimodal pattern, in which 50% of update intervals are smaller than 0.03 hours, while 25% of update intervals are larger than 24.0 hours. The reason of such a bimodal pattern in MSRC is that a volume is responsible for source control (i.e., *src1_0*) and updates data blocks daily. If we exclude the daily updates, most of the written blocks in MSRC have very short update intervals.

Figure 16 shows the boxplots of update intervals of different groups of percentiles across all volumes in AliCloud and MSRC. We see that the distributions of update intervals have high variations across volumes in both AliCloud and MSRC. For example, in AliCloud, the 50th percentiles of update intervals of all volumes range from 1 second to 17.8 days (Figure 16(a)), while in MSRC, the 50th percentiles of update intervals of all volumes range from 1 minute to 24 hours.

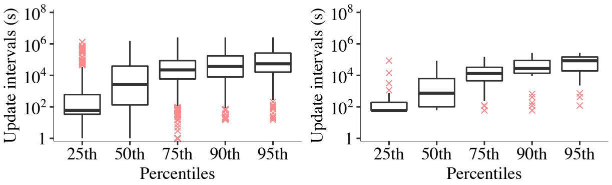
Many volumes have non-negligible proportions of short update intervals in their update requests. To further examine the distributions of update intervals in individual volumes, we divide the update intervals by duration into four groups: (i) less than 5 minutes, (ii) 5-30 minutes, (iii) 30-240 minutes, and (iv) more than 240 minutes. We calculate the proportions for the four groups of update intervals for each volume, and represent the proportions across all volumes by boxplots.

Figure 17 shows the boxplots of proportions for the four groups of update intervals across all volumes in both AliCloud and MSRC. Both AliCloud and MSRC have large proportions of either very small or very large update intervals. In AliCloud, half of the volumes have more than 35.2% and 38.2% of update intervals in less than 5 minutes and in more than 240 minutes, respectively (Figure 17(a)), while in MSRC, half of the volumes have more than 47.2% and 18.9% of update intervals in less

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.

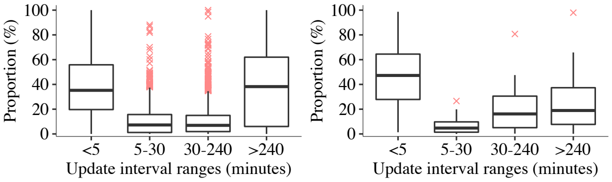
Percentiles (hours)	25th	50th	75th	90th	95th
AliCloud	0.03	1.59	15.5	50.3	120.2
MSRC	0.02	0.03	24.0	24.0	24.1

TABLE VI: Finding 14. Overall percentiles of update intervals across all volumes.



(a) AliCloud (b) MSRC

Fig. 16: Finding 14. Boxplots of percentiles of update intervals across all volumes.



(a) AliCloud (b) MSRC

Fig. 17: Finding 14. Boxplots of proportions for the four groups of update intervals across all volumes.

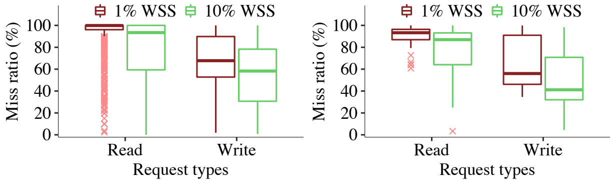
than 5 minutes and in more than 240 minutes (Figure 17(b)). Thus, a substantial amount of data is either updated frequently or not updated for long.

Finding 15. *Some volumes in AliCloud have low miss ratios even under a small cache size. Also, AliCloud shows higher reduction in miss ratios than MSRC when the cache size increases.*

Finally, we study the impact of caching with respect to the temporal patterns of the volumes. For each volume, we simulate a fixed-size cache for both reads and writes using the LRU policy, and evaluate the corresponding cache miss ratios for reads and writes. Here, we select 1% and 10% of the WSS of a volume as the cache size.

Figure 18 shows the boxplots of miss ratios across all volumes in both AliCloud and MSRC. Some volumes show low miss ratios (i.e., LRU-based caching is effective). For the cache size of 10% of WSS, the 25th percentiles of the miss ratios for reads and writes are 59.4% and 30.7%, respectively, in AliCloud (Figure 18(a)), while the corresponding miss ratios are 64.1% and 32.0%, respectively, in MSRC (Figure 18(b)). Also, some volumes in AliCloud can have very low miss ratios when the cache size is only 1% of WSS, implying that the access patterns of such volumes have high temporal locality.

AliCloud has higher reduction in miss ratios when the cache size increases from 1% to 10% of WSS. In AliCloud, the 25th percentiles of the miss ratios for reads and writes reduce from 96.1% to 59.4% and from 52.8% to 30.7% (i.e., 36.7% and 22.1% of absolute reduction), respectively (Figure 18(a)), while in MSRC, the 25th percentiles of the miss ratios for reads and writes reduce from 86.9% to 64.1% and from 46.2% to 32.1% (i.e., 22.8% and 14.1% of absolute reduction), respectively (Figure 18(b)). The higher reduction may be also attributed to the higher temporal locality for the volumes in AliCloud.



(a) AliCloud (b) MSRC

Fig. 18: Finding 15. Boxplots of miss ratios for reads and writes across all volumes, under the cache sizes of 1% and 10% of the WSS of a volume.

V. Summary of Findings

We now discuss the implications of our findings of the trace analysis in both AliCloud and MSRC. We show how the findings address the design considerations in cloud block storage, including load balancing, cache efficiency, and storage cluster management (Section II-B).

Load balancing. We focus on the average and peak intensities as well as the activeness of volumes. From Finding 1, we observe that while many applications are hosted in the cloud, the volumes in cloud block storage have similar load intensities to those in traditional data centers more than a decade ago.

From Findings 2-4, we observe the existence of burstiness in a non-negligible fraction of volumes. While the overall burstiness remains low, the burstiness can be severe in individual volumes, thereby leading to performance degradations if load balancing is not properly maintained. Both the high diversity of workloads and the presence of bursty requests make the load balancing of cloud block storage more challenging than in traditional data centers.

From Findings 5-7, we observe that writes are the dominant factor of activeness, while a large number of volumes are not active in reads. In particular, most volumes in cloud block storage are write-dominant (Section III-C). It is thus possible to offload writes (e.g., by redirecting writes to other storage locations) to create idle periods in cloud block storage workloads for power savings [21].

In the design of load balancing, the data placement strategies should be aware of the diversity of workloads and the burstiness of individual volumes. The log-structured design [22] is proven useful for balancing the write traffic in cloud-scale flash-based storage [32].

Cache efficiency. We study the spatial and temporal characteristics of volumes, which provide guidelines for motivating new caching designs for cloud block storage.

From Findings 9 and 15, we observe the patterns of both spatial and temporal traffic aggregations in a small fraction of blocks, especially for writes. Many volumes in cloud block storage show high aggregations of reads and writes, implying

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.

that it is viable to allocate limited cache resources for absorbing substantial amounts of reads and writes.

From Finding 10, we observe that many volumes in cloud block storage have reads and writes aggregated in read-mostly and write-mostly blocks, respectively. Thus, one possible caching admission policy is to identify the read-only and write-only blocks in workloads, as such blocks can absorb a substantial amount of I/O traffic.

From Findings 12 and 13, the blocks that have been written tend to be rewritten again, while the elapsed time for the next read to come is longer than the next write. In contrast, the blocks that have been read tend to receive another read or write after a long period of time. Thus, if our goal is to absorb writes with caching, a possible strategy is to favor the caching of the blocks that have been written rather than those that have been read, as the latter may unlikely generate write hits. Also, cloud block storage can benefit from disk-based write caching [24], due to the limited reads from the disk-based cache.

Storage cluster management. Characterizing the spatial and temporal characteristics of volumes is also critical for storage cluster management. Here, we focus on flash-based storage (Section II-A).

From Finding 8, we observe that upper-layer applications in cloud block storage issue lots of small and random I/Os, which are known to hurt both the performance and endurance of flash-based storage [19]. The log-structured storage design [22] and I/O clustering [19] can help mitigate the overhead of small and random I/Os.

From Findings 11 and 14, the update patterns have high variations across volumes, both spatially and temporally. The varying update patterns can harm the effectiveness of garbage collection and wear leveling in flash [10]. Thus, cloud block storage systems should take into account the varying patterns when optimizing update workloads for flash-based storage. A possible direction is to maintain the flash-translation layer (FTL) at the system level [8] to flexibly coordinate the I/Os issued to flash.

VI. Related Work

We review related work on the field studies on storage workloads and how they inspire storage system designs.

Characterization of storage workloads. Several field studies characterize storage workloads using block-level I/O traces in various architectures, such as virtual machines [3], Windows servers [11], [21], smartphone applications [35], containerized applications [9], and virtual desktop infrastructures [12]. In contrast, our field study focuses on cloud block storage that supports a diverse set of cloud applications in large-scale production. In particular, we provide findings and insights on performance optimizations for load balancing, caching efficiency, and storage cluster management.

Inspirations from load intensity. Some designs are inspired by the characteristics of load intensity in storage workloads. Narayanan et al. [21] offload writes to reduce power consumptions with the observation that some volumes are idle in reads, thereby removing writes in those volumes can increase the idle periods for power saving. SRCMap [26] reduces power consumptions using sampling and replication, based on the observation on the I/O size and intensity of active data sets. Ursa [13] adopts the log-structured design, based on the observation that small writes dominate in real-world workloads.

Inspirations from spatial patterns. Some designs exploit the spatial characteristics of storage workloads. BORG [5] organizes frequently written data in a small dedicated disk partition to reduce the I/O seek time. FlashTier [23] manages sparse address mappings in flash caching, as storage I/Os are often aggregated in a small number of blocks. ACGR [14] regulates I/O accesses for flash storage, based on the observation of read and write aggregations in read-only and write-only blocks, respectively. To improve the update performance in erasure-coded storage, CodFS [7] proposes dynamic reserved space management for parity updates to address the varying working sets of updates across storage workloads, while PBS [34] exploits the large fractions of overwrites to mitigate parity update overhead.

Inspirations from temporal patterns. Some designs exploit the temporal characteristics of storage workloads. Griffin [24] leverages the large time intervals between writes and the subsequent reads to the same block to build an HDD-based write cache for improving the SSD lifetime. Some studies leverage the characteristics of update intervals in storage workloads for improving write performance [15], lifetime [6], garbage collection modeling, and data reduction [33] in SSDs. Counter Stacks [31] and SHARDS [28] consider the reuse distance (i.e., the number of distinct items accessed between two accesses to the same item) to improve caching efficiency.

Cloud block storage systems. Several cloud block storage designs are proposed in the literature. Parallax [17] provides storage virtualization for virtual machines atop shared block storage. Blizzard [18] manages POSIX applications atop cloud block storage. Ursa [13] is a hybrid block storage system that combines HDDs (hard disk drives) and SSDs (solid-state drives) for cloud-scale virtual disks. PBS [34] supports erasure-coded cloud block storage with efficient updates. Our trace analysis provides suggestions for optimizing such cloud block storage designs.

VII. Conclusion

We present an in-depth trace analysis on a production cloud block storage system, using block-level I/O traces collected from Alibaba Cloud. We reveal the commonalities and differences from the existing public block-level I/O traces. We highlight 15 findings, based on which we further discuss the implications on three practical design considerations for cloud block storage, including load balancing, cache efficiency, and storage cluster management. We have released our traces for the community to identify new findings and research directions for storage system research.

Acknowledgments

This work was supported by Alibaba Group via the Alibaba Innovation Research (AIR) program.

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.

References

[1] Alibaba Cloud Block Storage. https://www.alibabacloud.com/help/doc-detail/63136.htm.

[2] Amazon EBS. https://aws.amazon.com/ebs/.

[3] I. Ahmad. Easy and efficient disk I/O workload characterization in vmware ESX server. In *Proc. of IEEE IISWC*, 2007.

[4] D. Arteaga, J. Cabrera, J. Xu, S. Sundararaman, and M. Zhao. Cloud-Cache: On-demand flash cache management for cloud computing. In *Proc. of USENIX FAST*, 2016.

[5] M. Bhadkamkar, J. Guerra, L. Useche, S. Burnett, J. Liptak, R. Rangaswami, and V. Hristidis. BORG: Block-reORGanization for self-optimizing storage systems. In *Proc. of USENIX FAST*, 2009.

[6] Y. Cai, Y. Luo, E. F. Haratsch, K. Mai, and O. Mutlu. Data retention in MLC NAND flash memory: Characterization, optimization, and recovery. In *Proc. of IEEE HPCA*, 2015.

[7] J. C. W. Chan, Q. Ding, P. P. C. Lee, and H. H. W. Chan. Parity logging with reserved space: Towards efficient updates and recovery in erasure-coded clustered storage. In *Proc. of USENIX FAST*, 2014.

[8] T.-c. Chiueh, W. Tsao, H.-C. Sun, T.-F. Chien, A.-N. Chang, and C.-D. Chen. Software orchestrated flash array. In *Proc. of ACM SYSTOR*, 2014.

[9] T. Harter, B. Salmon, R. Liu, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. Slacker: Fast distribution with lazy docker containers. In *Proc. of USENIX FAST*, 2016.

[10] J. He, S. Kannan, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. The unwritten contract of solid state drives. In *Proc. of ACM EuroSys*, 2017.

[11] S. Kavalanekar, B. Worthington, Q. Zhang, and V. Sharda. Characterization of storage workload traces from production windows servers. In *Proc. of IEEE IISWC*, 2008.

[12] C. Lee, T. Kumano, T. Matsuki, H. Endo, N. Fukumoto, and M. Sugawara. Understanding storage traffic characteristics on enterprise virtual desktop infrastructure. In *Proc. of ACM SYSTOR*, 2017.

[13] H. Li, Y. Zhang, D. Li, Z. Zhang, S. Liu, P. Huang, Z. Qin, K. Chen, and Y. Xiong. URSA: Hybrid block storage for cloud-scale virtual disks. In *Proc. of ACM EuroSys*, 2019.

[14] Q. Li, L. Shi, C. J. Xue, K. Wu, C. Ji, Q. Zhuge, and E. H.-M. Sha. Access characteristic guided read and write cost regulation for performance improvement on flash memory. In *Proc. of USENIX FAST*, 2016.

[15] R. S. Liu, C. L. Yang, and W. Wu. Optimizing NAND flash-based SSDs via retention relaxation. In *Proc. of USENIX FAST*, 2012.

[16] S. Liu, S. Wang, Q. Cao, Z. Lu, H. Jiang, J. Yao, Y. Dong, and P. Yang. Analysis of and optimization for write-dominated hybrid storage nodes in cloud. In *Proc. of ACM SoCC*, 2019.

[17] D. T. Meyer, G. Aggarwal, B. Cully, G. Lefebvre, M. J. Feeley, N. C. Hutchinson, and A. Warfield. Parallax: Virtual disks for virtual machines. In *Proc. of ACM EuroSys*, 2008.

[18] J. Mickens, E. B. Nightingale, J. Elson, K. Nareddy, D. Gehring, B. Fan, A. Kadav, V. Chidambaram, and O. Khan. Blizzard: Fast, cloud-scale block storage for cloud-oblivious applications. In *Proc. of USENIX NSDI*, 2014.

[19] C. Min, K. Kim, H. Cho, S.-W. Lee, and Y. I. Eom. SFS: Random write considered harmful in solid state drives. In *Proc. of USENIX FAST*, 2012.

[20] A. K. Mishra, J. L. Hellerstein, W. Cirne, and C. R. Das. Towards characterizing cloud backend workloads: Insights from google compute clusters. In *Proc. of ACM SIGMETRICS*, 2010.

[21] D. Narayanan, A. Donnelly, and A. Rowstron. Write Off-Loading: Practical power management for enterprise storage. In *Proc. of USENIX FAST*, 2008.

[22] M. Rosenblum and J. K. Ousterhout. The design and implementation of a log-structured file system. *ACM Trans. on Computer Systems*, 10(1):26–52, 1992.

[23] M. Saxena, M. M. Swift, and Y. Zhang. FlashTier: a lightweight, consistent and durable storage cache. In *Proc. of ACM EuroSys*, 2012.

[24] G. Soundararajan, V. Prabhakaran, M. Balakrishnan, and T. Wobber. Extending SSD lifetimes with disk-based write caches. In *Proc. of USENIX FAST*, 2010.

[25] M. Tarihi, H. Asadi, and H. Sarbazi-Azad. DiskAccel: Accelerating disk-based experiments by representative sampling. In *Proc. of ACM SIGMETRICS*, 2015.

[26] A. Verma, R. Koller, L. Useche, and R. Rangaswami. SRCMap: Energy proportional storage using dynamic consolidation. In *Proc. of USENIX FAST*, 2010.

[27] M. Wajahat, A. Yele, T. Estro, A. Gandhi, and E. Zadok. Distribution fitting and performance modeling for storage traces. In *Proc. of IEEE MASCOTS*, pages 138–151. IEEE, 2019.

[28] C. A. Waldspurger, N. Park, A. Garthwaite, and I. Ahmad. Efficient MRC construction with SHARDS. In *Proc. of USENIX FAST*, 2015.

[29] H. Wang, X. Yi, P. Huang, B. Cheng, and K. Zhou. Efficient SSD caching by avoiding unnecessary writes using machine learning. In *Proc. of ACM ICPP*, 2018.

[30] S. Wang, Z. Lu, Q. Cao, H. Jiang, J. Yao, Y. Dong, and P. Yang. BCW: Buffer-controlled writes to hdds for ssd-hdd hybrid storage server. In *Proc. of USENIX FAST*, 2020.

[31] J. Wires, S. Ingram, Z. Drudi, N. J. A. Harvey, and A. Warfield. Characterizing storage workloads with counter stacks. In *Proc. of USENIX OSDI*, 2014.

[32] E. Xu, M. Zheng, F. Qin, Y. Xu, and J. Wu. Lessons and actions: What we learned from 10k SSD-related storage system failures. In *Proc. of USENIX ATC*, 2019.

[33] J. Yang, S. Pei, and Q. Yang. WARCIP: Write amplification reduction by clustering I/O pages. In *Proc. of ACM SYSTOR*, 2019.

[34] Y. Zhang, H. Li, S. Liu, J. Xu, and G. Xue. PBS: An efficient erasure-coded block storage system based on speculative partial writes. In *ACM Transactions on Storage*, 2020.

[35] D. Zhou, W. Pan, W. Wang, and T. Xie. I/O characteristics of smartphone applications and their implications for eMMC design. In *Proc. of IEEE IISWC*, 2015.

Authorized licensed use limited to: University of Canberra. Downloaded on May 21,2021 at 01:39:08 UTC from IEEE Xplore. Restrictions apply.