Reducing Write Traffic to Remote Storage

Ramnatthan Alagappan, Brandon Davis, Avinaash Gupta

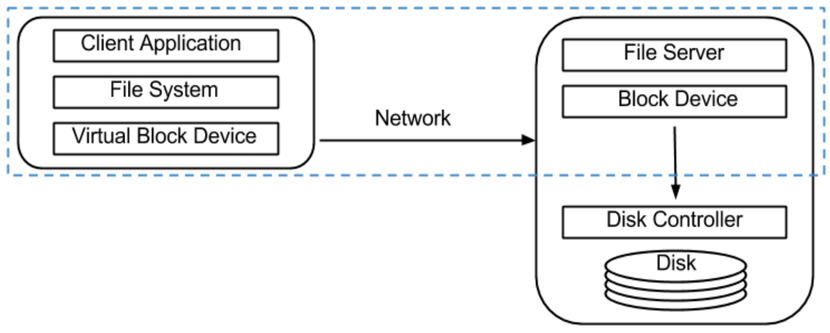
WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

IS BLOCK INTERFACE APPROPRIATE FOR REMOTE STORAGE?

The block device interface enables computer system software such as file systems to communicate with the storage system. When dealing with sector-based storage devices like HDDs, it is necessary to work with blocks of data. But many protocols maintain this block granularity for data in other layers of the system. This adds significant overhead in the amount of data transferred to the storage system. In context of remote storage, the problem becomes more important because the data is transferred over the network. For example, Network Block Device (NBD), and iSCSI work with blocks in the network layer, sending entire blocks of data for each request. This can result in excessive data transfer when the useful data in a request is smaller than the block size. In congested networks, every byte counts.

With small modifications to NBD to avoid full block requests where possible, we show a savings of over 70% in write traffic for certain workloads.

LAYERS IN A REMOTE STORAGE



DESIGN AND IMPLEMENTATION OF OUR PROTOTYPE

CACHE AND DIFF

Key idea: Logically partition the block into ***chunks*** of equal size. Modified chunks are identified by a simple diff. A bitmap is also sent along with the chunks to denote which all chunks have been modified.
Observation: Choice of chunk size and bitmap size will have effects on savings achieved.
Client:

- Cache blocks which are read/written through NBD.
- For subsequent writes, perform diff and send only the changed chunks of the block(s).

Server:

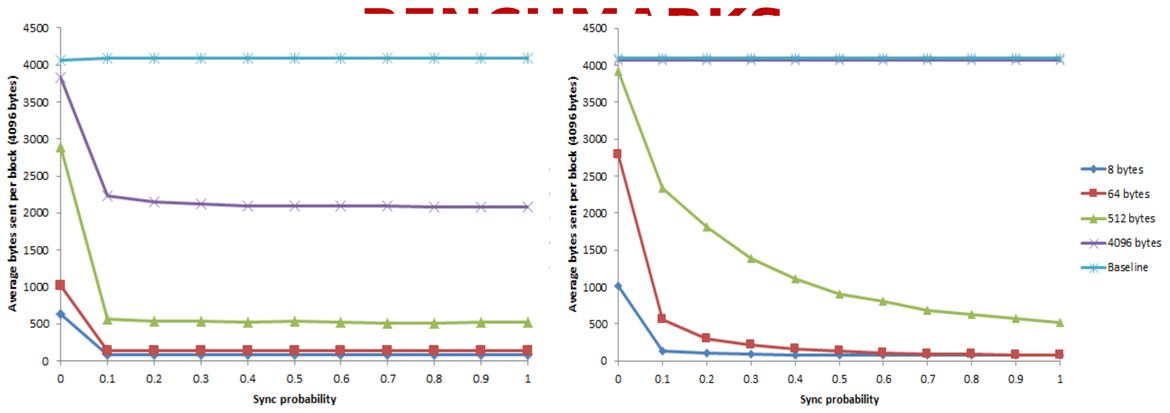
- Read the blocks(s) from disk, apply diff and write back.

CHOICE OF CHUNK SIZE

- The chosen chunk size and the corresponding bitmap size will influence the effectiveness of our technique.
- While 1 byte chunks give finest granularity for finding unmodified chunks, we need to find 513 such chunks to benefit from our technique.
- For 32 byte chunks we just need to find 1 unmodified chunk for savings.

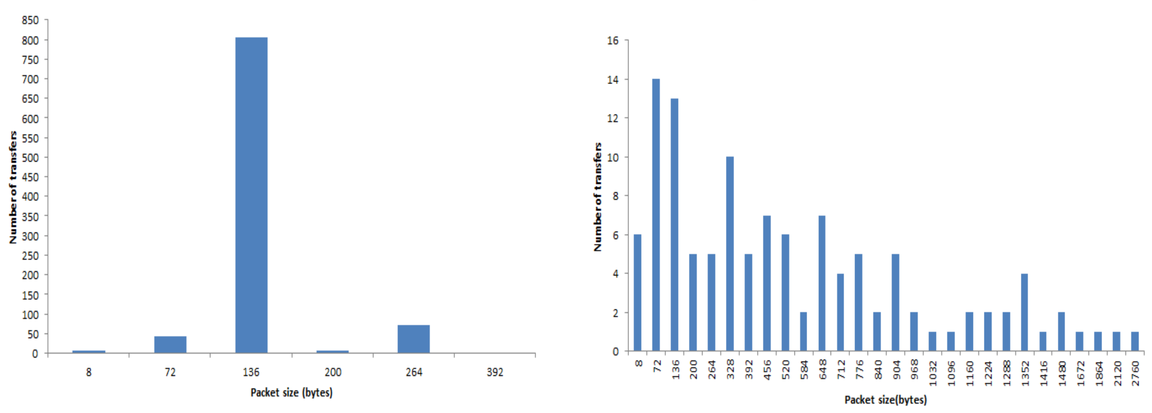
Chunk size (bytes)1	Bitmap size (bytes)	For savings
4096	1	1
2048	1	1
1024	1	1
512	1	1
256	2	1
128	4	1
64	8	1
32	16	1
16	32	3
8	64	9
4	128	33
2	256	129
1	512	513

MICRO BENCHMARKS



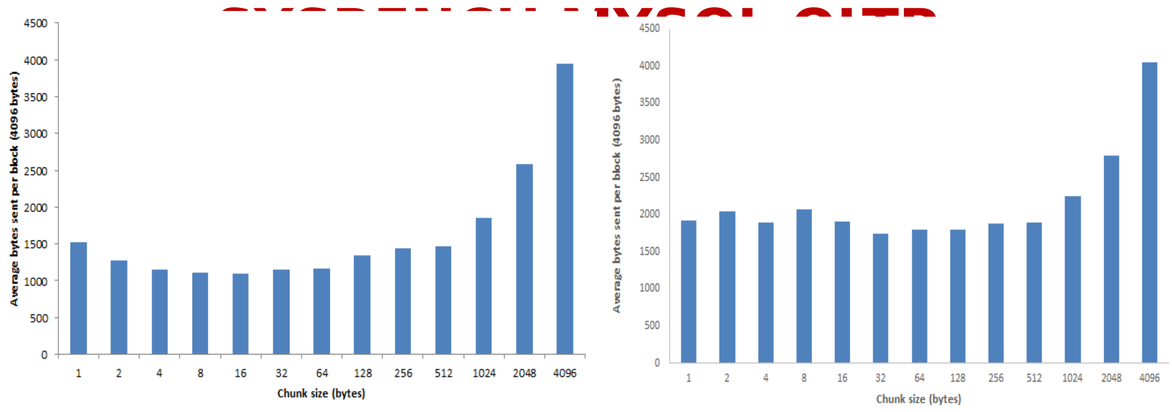
Random writes

Sequential writes



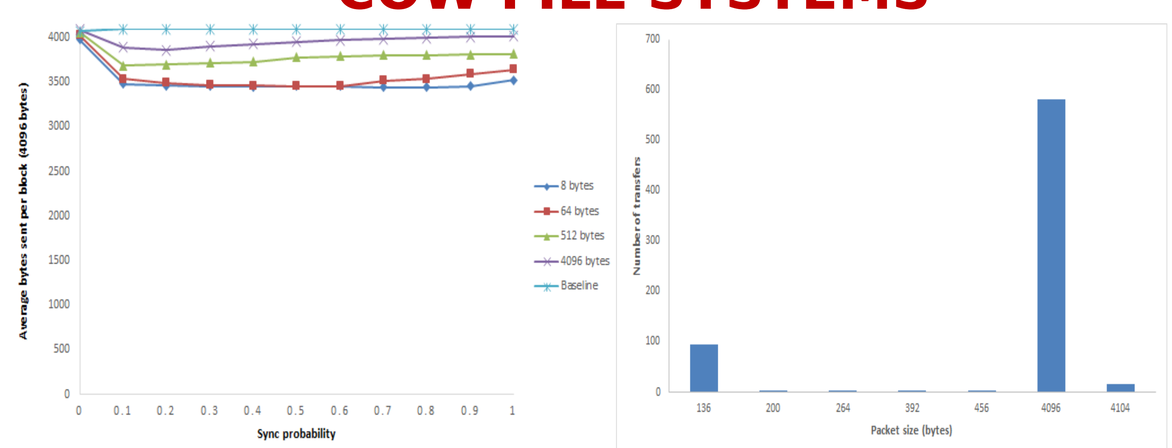
Distribution of write sizes for 64 byte writes with sync probability of 0.1

MACRO BENCHMARKS-



Average bytes sent per block– without and with cache (LRU eviction)

COW FILE SYSTEMS



FUTURE WORK

- Study proprietary remote block device services such as Amazon EBS.
- We want to answer questions like:
  - How multiple clients connect to the device in such a scenario?
  - How can we ensure consistency across all clients?
  - How to SCSI reservations fit in this model?
- Can we envision a file system which is divided into block independent and block dependent layers?
- Study the impact of savings in network traffic that our technique brings about in congested networks.

REFERENCES

[1] Anurag Acharya, Mustafa Uysal, Joel Saltz, Active disks: programming model, algorithms and evaluation, ACM SIGPLAN Notices, v.33 n.11,p.81-91, Nov. 1998.
[2] Riedel, E. and Gibson, G., Active Disks - RemoteExecution for Network-Attached Storage, Technical Report CMU-CS-97-198, December 1997.
[3] S. Boboila, Y. Kim, S. S. Vazhkudai, P. Desnoyers, and G. M. Shipman, Active Flash: Out-of-core Data Analytics on Flash Storage, In MSST,2012.
[4] M. Factor, K. Meth, D. Naor, O. Rodeh, and J. Satran., Object Storage: the future building block for storage systems, In LGDI 05: Proceed-ings of the 2005 IEEE International Symposiumon Mass Storage Systems and Technology, pp.119123, Washington, DC, USA, 2005.
[5] Matias Bjorling, Philippe Bonnet, Luc Bouganim,Niv Dayan, et al., The necessary death of the block device interface, In 6th Biennial Conference on Innovative Data Systems Research (CIDR), 2013.