Native SSD Cache for Lustre OSTs

DataDirect Networks, Inc.

Sebastien Buisson sbuisson@ddn.com

2016/06

Why SSD cache for Lustre?

- **Memory cache can help but the cache space is very limited.**
- **SSD has better performance than HDD but not always affordable for massive usage.**
- **SSD cache is able to accelerate applications with larger data sets.**

- **Solution:**
  - Cache + Access locality = Lower cost + Better performance

Why implement SSD cache on OSS side?

- **SSD cache is not necessary for MDS, since SSDs can be used as storage for MDTs.**

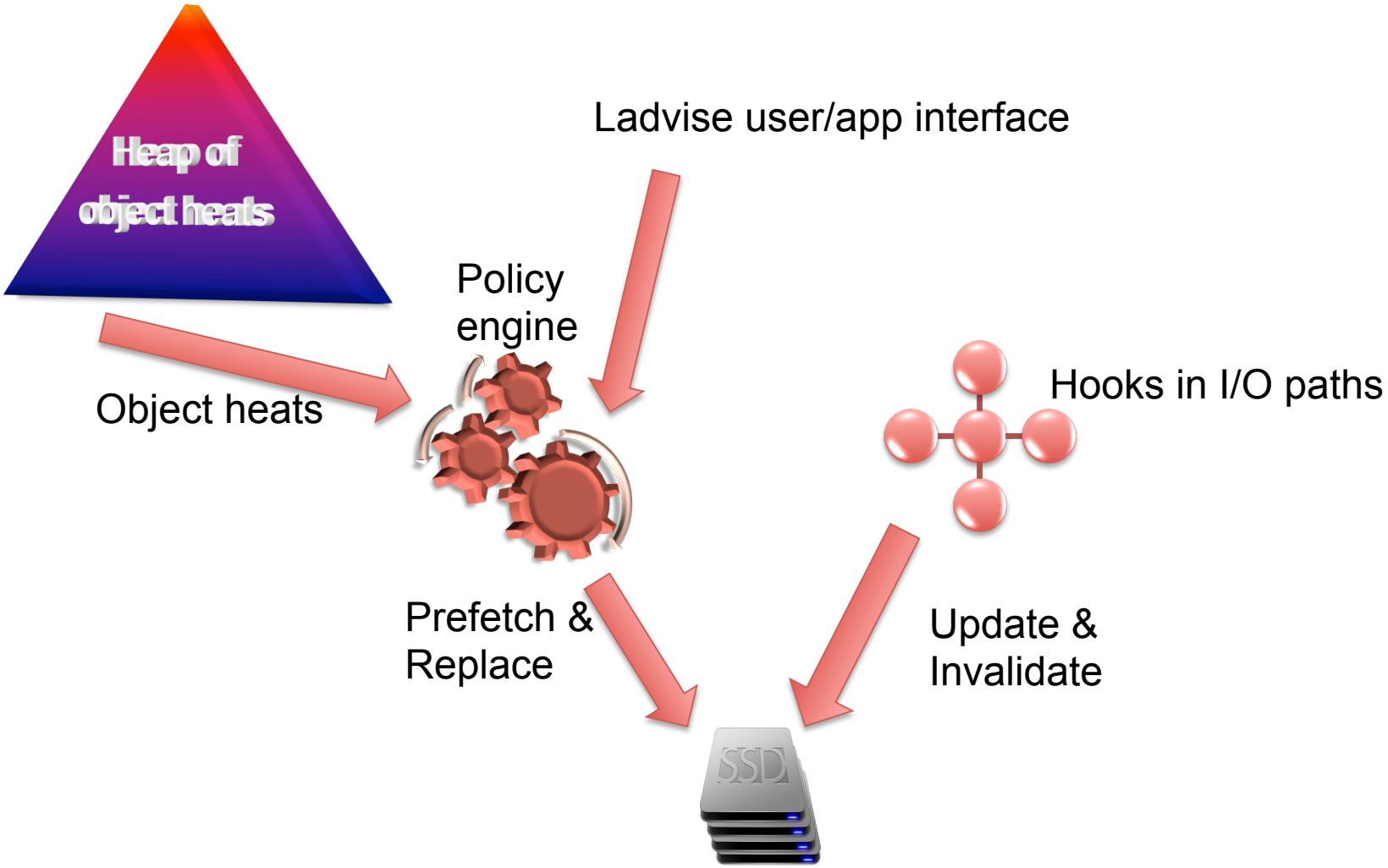
- **Hybrid drives for OST devices?**
  - Some hybrid drives lack APIs for cache management from Lustre side.
  - Cache management APIs are usually different between vendors.
- **OST pools made of SSDs?**
  - Transparent data migration and space management between pools is difficult.
- **SSD cache on OSS side!**
  - We implemented a cache system on OSS named **L2RC** (**Lustre Level 2 Read Cache**)

Design of L2RC

- **Currently focus on read-only Cache**
  - Write support will be added after read cache support is finished: **L2RC -> L2C**

- **Space allocation**
  - All space is divided into 1M buckets.
  - Bucket is the unit of space management.
  - A bucket contains multiple extents.
  - Extents size is between 4096 and 1M.
  - Cached data of an object is divided to multiple 1M extents and one variable-length extent

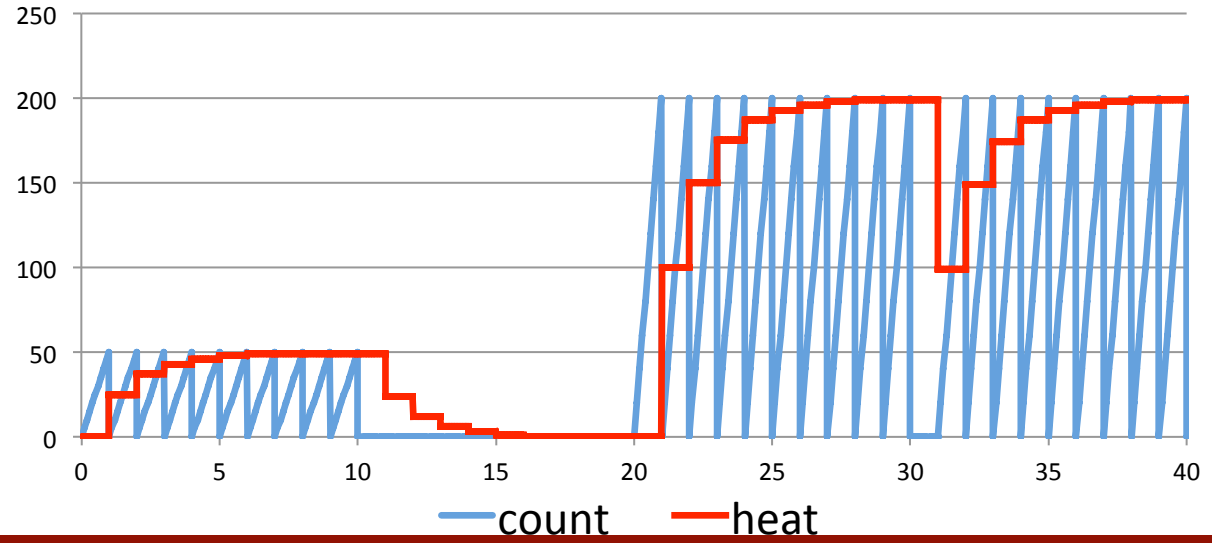
Space management of L2RC



Design of file heat

- **Lustre file heat is a relative attribute which can reflect the access frequency of the file/object.**
  - It grows as file is accessed, yet dissipates as time moves on.
- **`Heat[i+1]=`**

```
(1-Replacement_Percentage) x Heat[i]   +
Replacement_Percentage     x Acces[i]
```



Implementation of File Heat

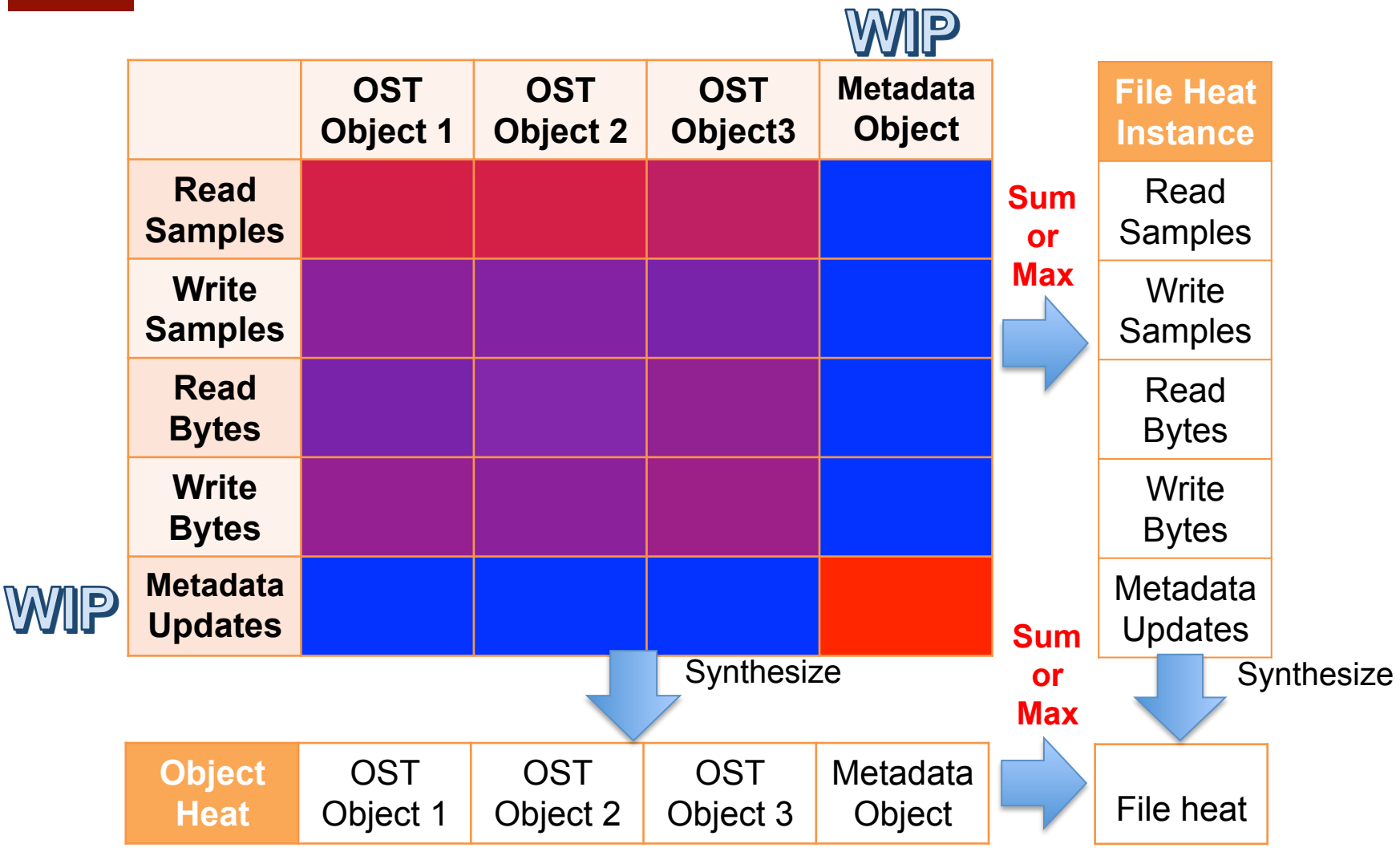
- **File Heat is stored in a heat map in memory.**
  - Plus one xattr on disk when necessary.

- **File Heat changes every period**
- **BUT very low impact on performance**
- **No background thread to scan objects**
  - File Heat is not even updated every period
    - Only when file is accessed
    - Or when file heat is queried
  - Heat xattr is not updated every period

- **Low memory footprint**
  - 152 bytes per object in OST

  ⇒ 150 MB for one million objects on an OSS

Implementation of file heat

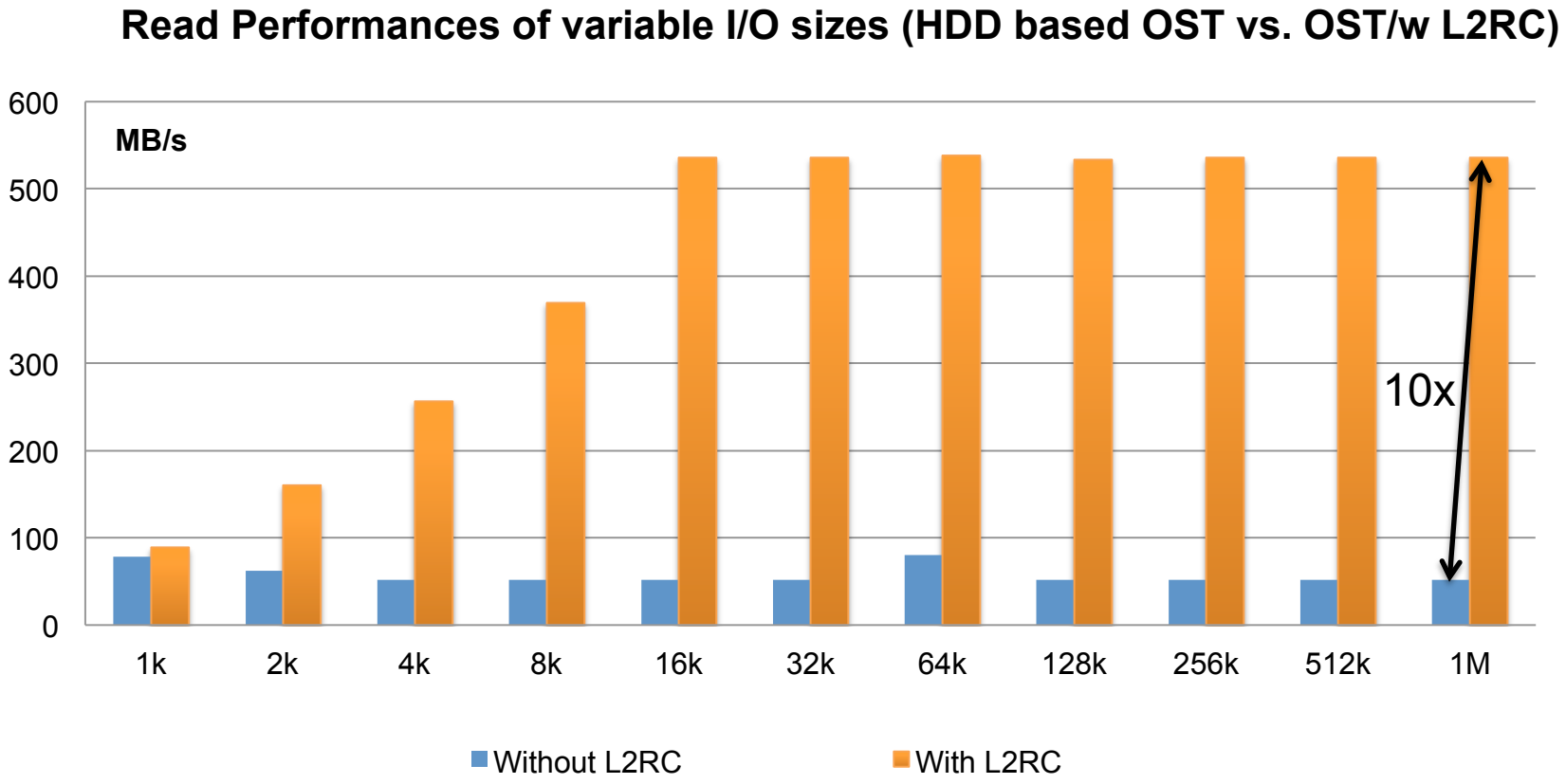


Prefetch based on ladvise

- **Ladvise**
  - A framework of Lustre to send advices/hints from clients to server components.
- **Ladvise is integrated into Lustre stacks**
  - Transparently handle file stripe of Lustre.
  - Give hints through the I/O path to keep efficient
- **Utility and API is simple to use**
  - “lfs fadvise” command to give advices on Lustre files
  - ioctl(LL_IOC_FADVISE) for advices from smart applications
- **Ladvise enables applications and users with external knowledge to intervene in cache management**

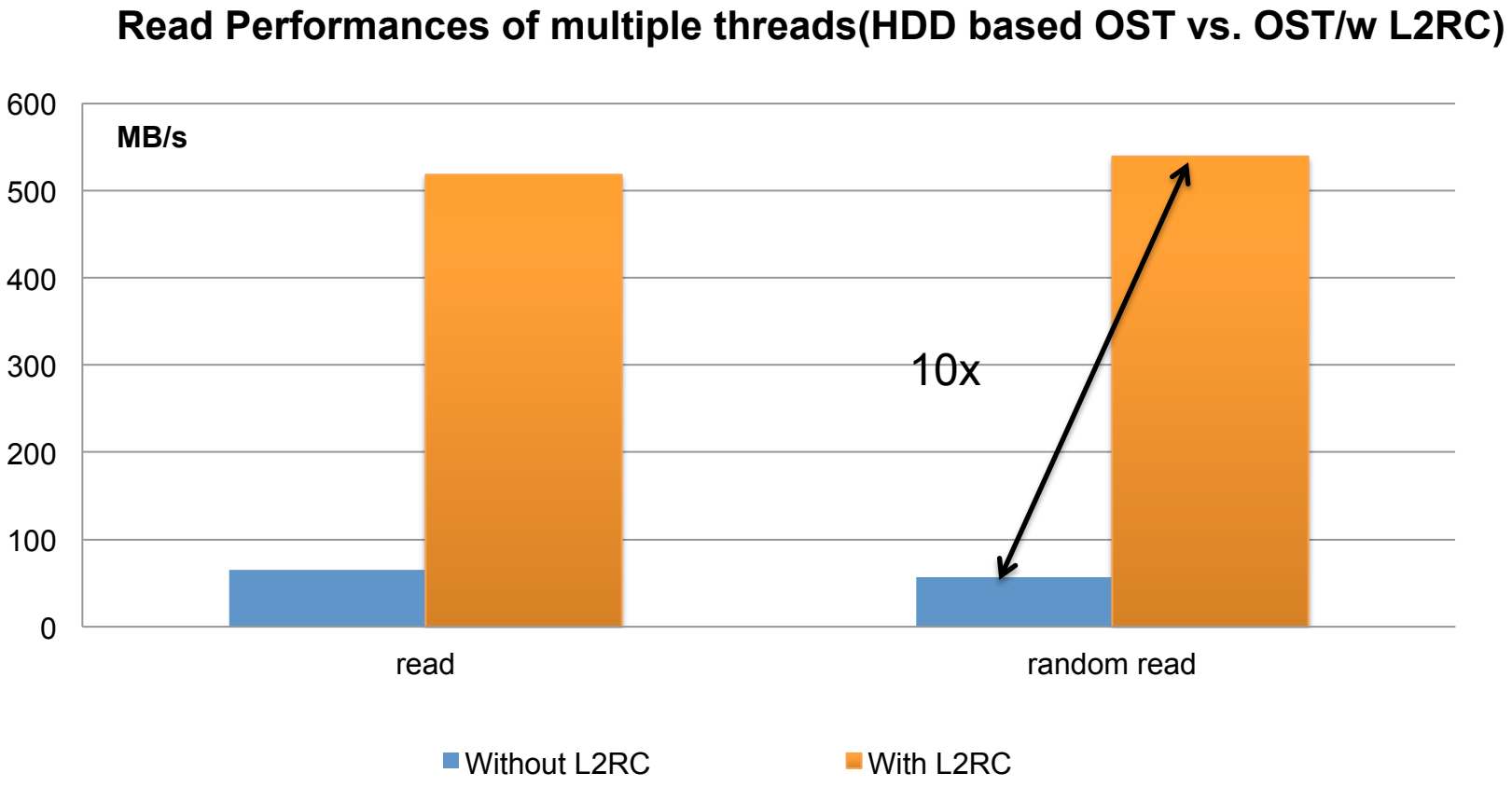
Benchmark results (1/3)

- **One big file of 120GB, single thread (dd)**



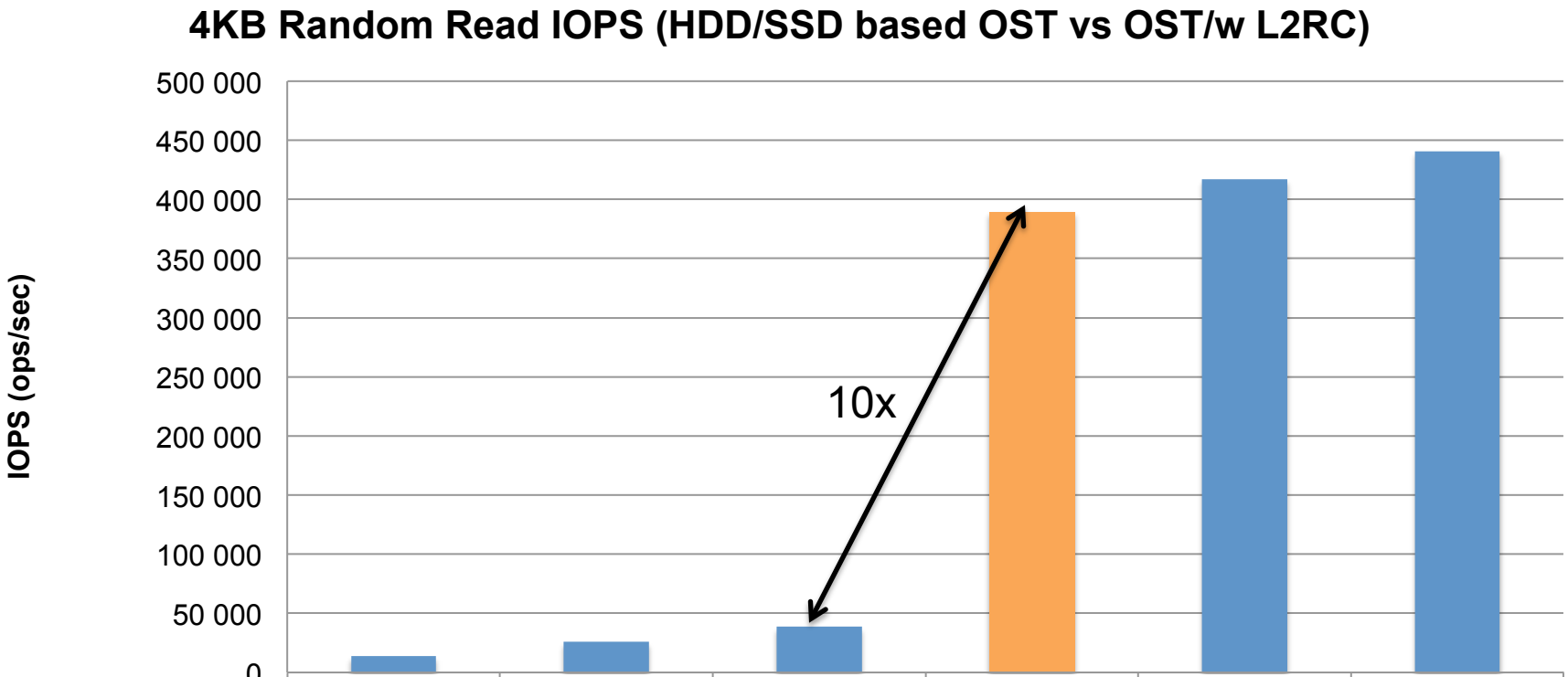
Benchmark results (2/3)

- **Multiple thread performance (tiobench with 4 threads)**



Benchmark results (3/3)

- **IOPS of multiple clients (IOR with 32 clients)**



	40 x Spindle Drive	80 x Spindle Drive	160 x Spindle Drive	160 x Spindle Drive with L2RC	4 x OST on SSD	4 x SSD(RAID1) Raw device
4K Random IOPS	13 739	26 064	38 688	389 232	416 994	440 428

Conclusion

- We implemented an SSD cache named L2RC on Lustre OSS.
- We provided automatic cache management mechanism based on file heat, as well as APIs based on ladvise.
- We proved that L2RC is able to present the maximum performance of SSD to applications.
- We demonstrated that L2RC might be able to accelerate read performance of different applications

Thank You!

Keep in touch with us

Team-jpsales@ddn.com

@ddn_limitless

company/datadirect-networks

102-0081
東京都千代田区四番町6-2
東急番町ビル 8F

TEL:03-3261-9101
FAX：03-3261-9140