

# 25,000のベンチマーク結果からわかった！ **OpenStack on Cephの性能**

Ver. 1.1

## OSCA技術検討会 OpenStack分科会

デル株式会社 日比野 正慶

レッドハット株式会社 佐々木 宏忠

株式会社日立ソリューションズ 工藤 雄大

株式会社日立ソリューションズ 平原 一帆



# 1. はじめに

## - OSCAのご紹介と本検証のねらい

デル株式会社

エンタープライズソリューション & アライアンス  
エンタープライズ テクノロジスト

日比野 正慶



# OSCA のご紹介



2012年2月にインテルとデルが中心となり発足、2016年6月時点で23社が参加。  
先進的且つ標準的な技術の情報発信を通じ、参画企業のビジネス活性化を目的とする。  
2016年、OSCA v2.0 で、ソリューションスコープを拡張、IOT実現に向けた取り組みを強化。



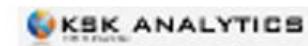
vmware®

BROCADE®

NTT DATA



cloudera



CITRIX®



Microsoft

HITACHI  
Inspire the Next  
日立ソリューションズ



五十音順

# OSCA の活動

- 各分科会による検証と、ブログ、ホワイトペーパーを通じた**情報発信**
- ビジネスにつなげるための、**各種セミナー主催やイベント参加**
- お客様やパートナー様との関係を強化する**技術交流会やパーティーイベントの開催**



# OSCA 3つのテーマ

ビッグデータ解析  
オープンネットワーキング  
オープンクラウドコンピューティング

## 現在進行中のプロジェクト

- **Ceph on OpenStack の性能評価**
- Docker 利用時の I/O 性能評価
- NFV性能評価 - OpenStack NFVI基盤上での vCPE性能評価
- ホワイトボックススイッチを利用したネットワークOS比較検証
- “Dell Bigdata / IoTラボ” POC/デモ環境構築

隨時新規プロジェクト受付中



# 検証の思惑と狙い



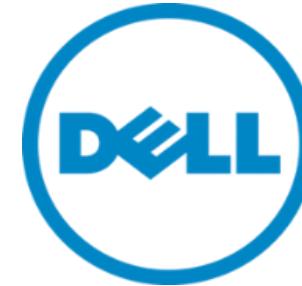
**OSCA OpenStack  
技術分科会リーダーとして  
テーマを検討**

Cephってどうなの？



**最新 RHEL OSP 8  
Cephがライセンスフリーに！**

Cephを使ってほしい



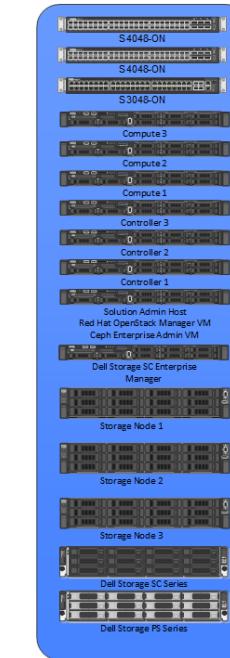
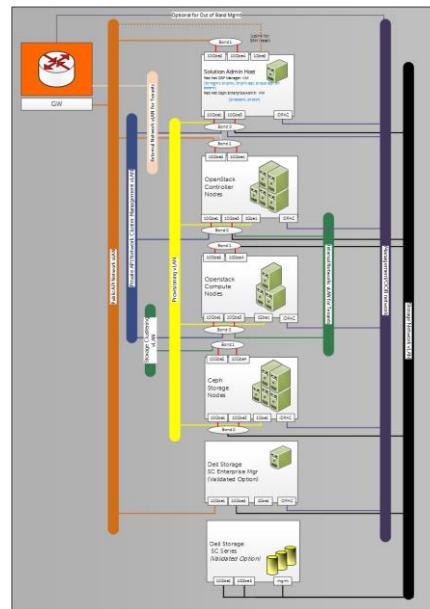
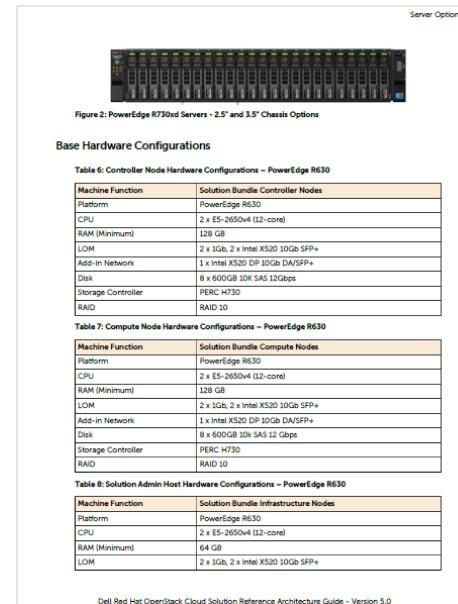
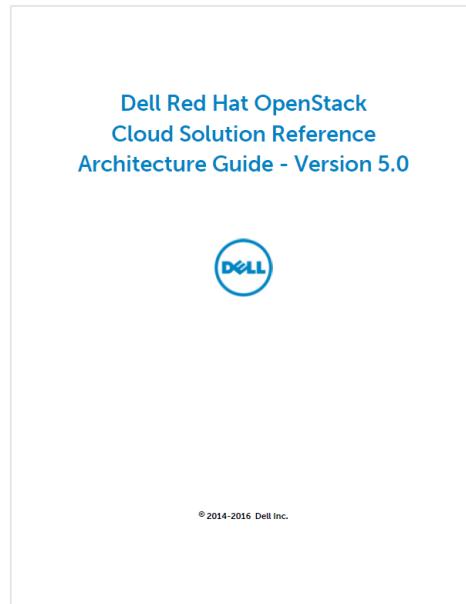
**最新OpenStack  
RAIリリース  
ストレージはCephおし**

案件が増えてきた

**Cephの情報が少ない OSCAでやろう！**

# デルの最新 OpenStack RA

- 2016年6月リリース 最新 OpenStack RA (RHEL OSP8対応)
- Director, Instance HA, OpenShift v3.3, Ceph v1.31, Midonet サポート
- 興味あるかたは、下記URLまたはデルブースまで



Download  
<http://www.dell.com/en-us/work/learn/assets/shared-content~data-sheets~en/documents~dell-red-hat-cloud-solutions.pdf>

## 2. 検証環境のご説明

レッドハット株式会社  
テクニカル・セールス本部  
ソリューションアーキテクト  
佐々木 忠弘

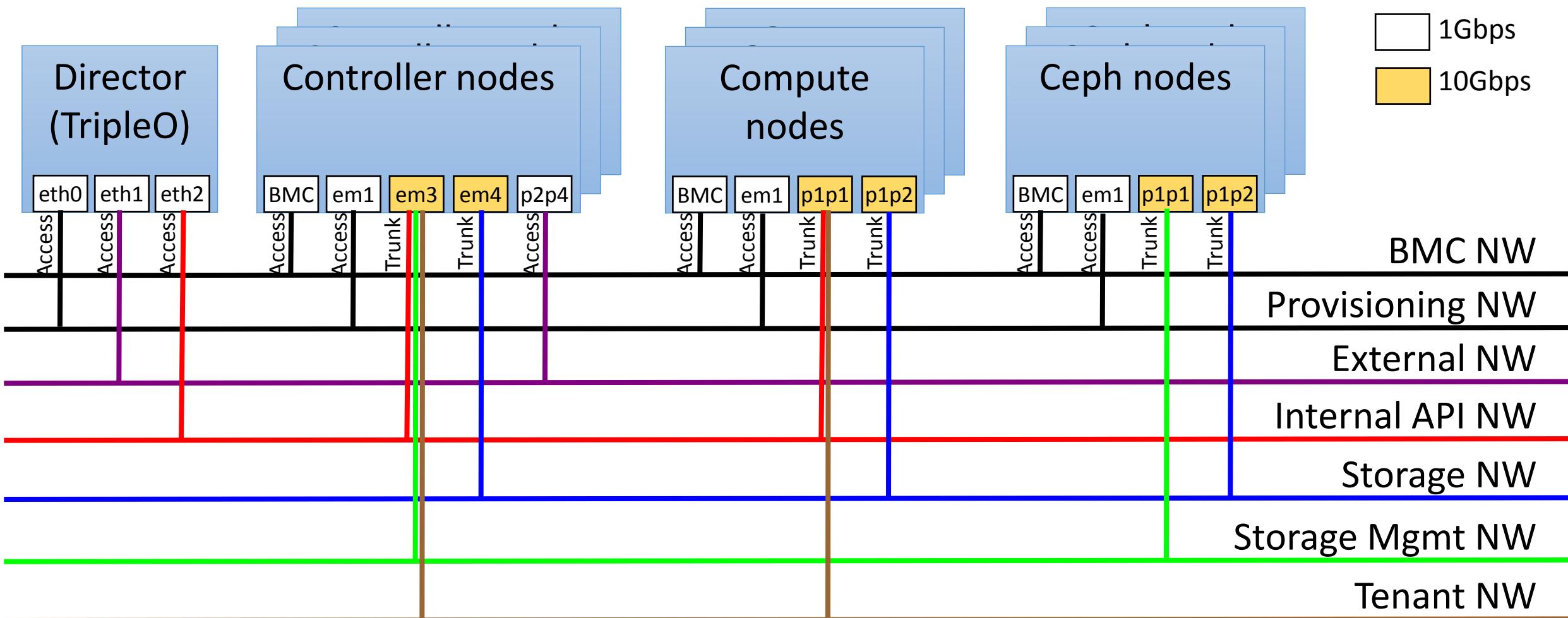


## 2.1 検証構成の特徴

- 一般的な機器
  - Dell PowerEdge C6220 (Xeon E5-2620×2(2GHz, 6Core))
  - HDDは10,000rpmのSAS、SSDは非NVMe
- 標準的な構成・設定、**特別なチューニング無し**
  - ほぼインストールガイドの設定例とおり
  - ほぼチューニングはしていない
- ソフトウェアのバージョン
  - Red Hat OpenStack Platform 7 (Kilo (RHEL7.2))
  - Red Hat Ceph Storage 1.3 (Hammer (RHEL7.2))

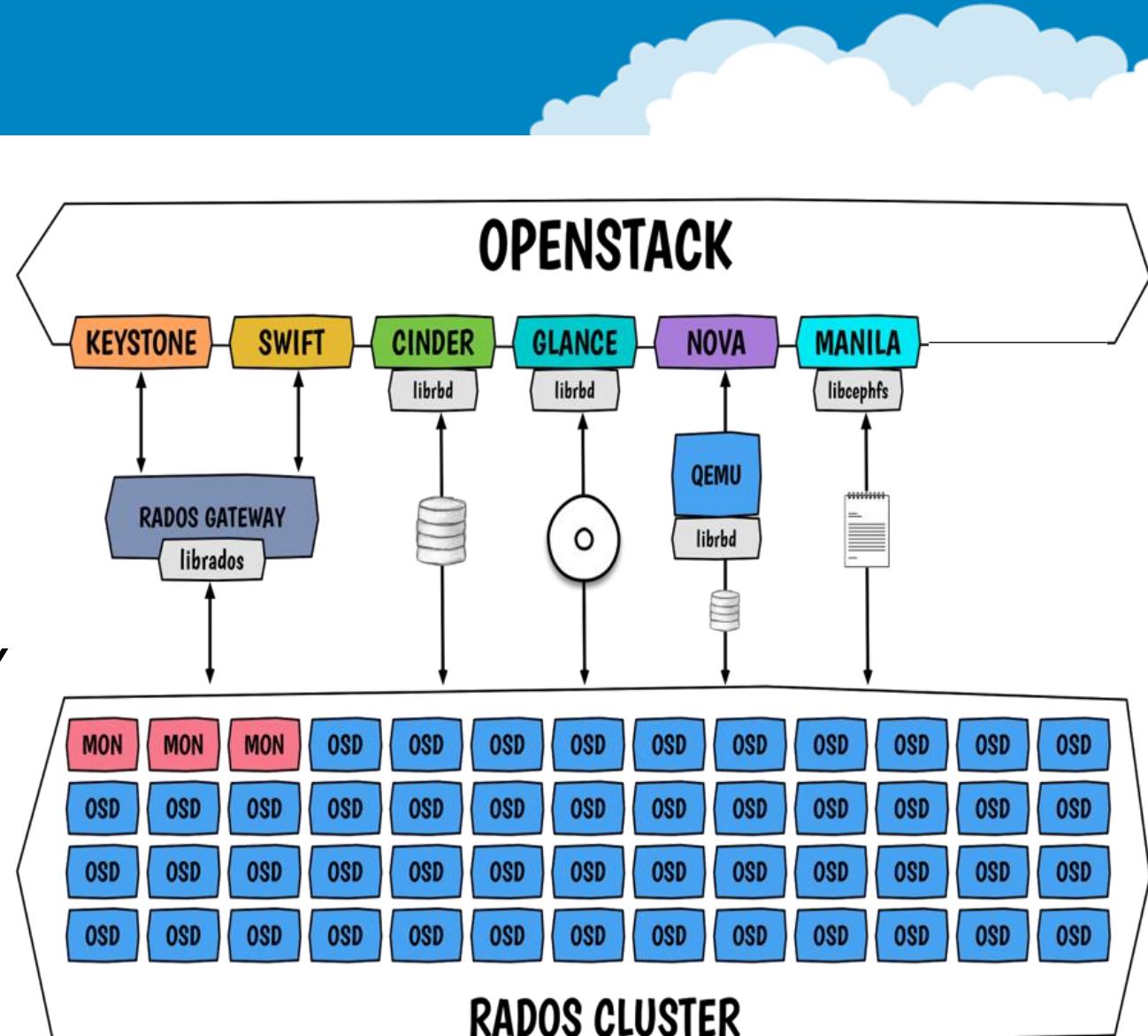
条件を変えた時に、  
どのように性能が  
変わるかがポイント

## 2.2 OpenStackの構成



## 2.3 Cephの特徴

- ・オープンソースのSoftware Defined Storage
- ・OpenStackのストレージバックエンドをCephに統合
- ・OPENSTACK USER SURVEYで、59%のユーザがCinderバックエンドに使用（※）
- ・CRUSHアルゴリズムにより大規模スケールアウト対応

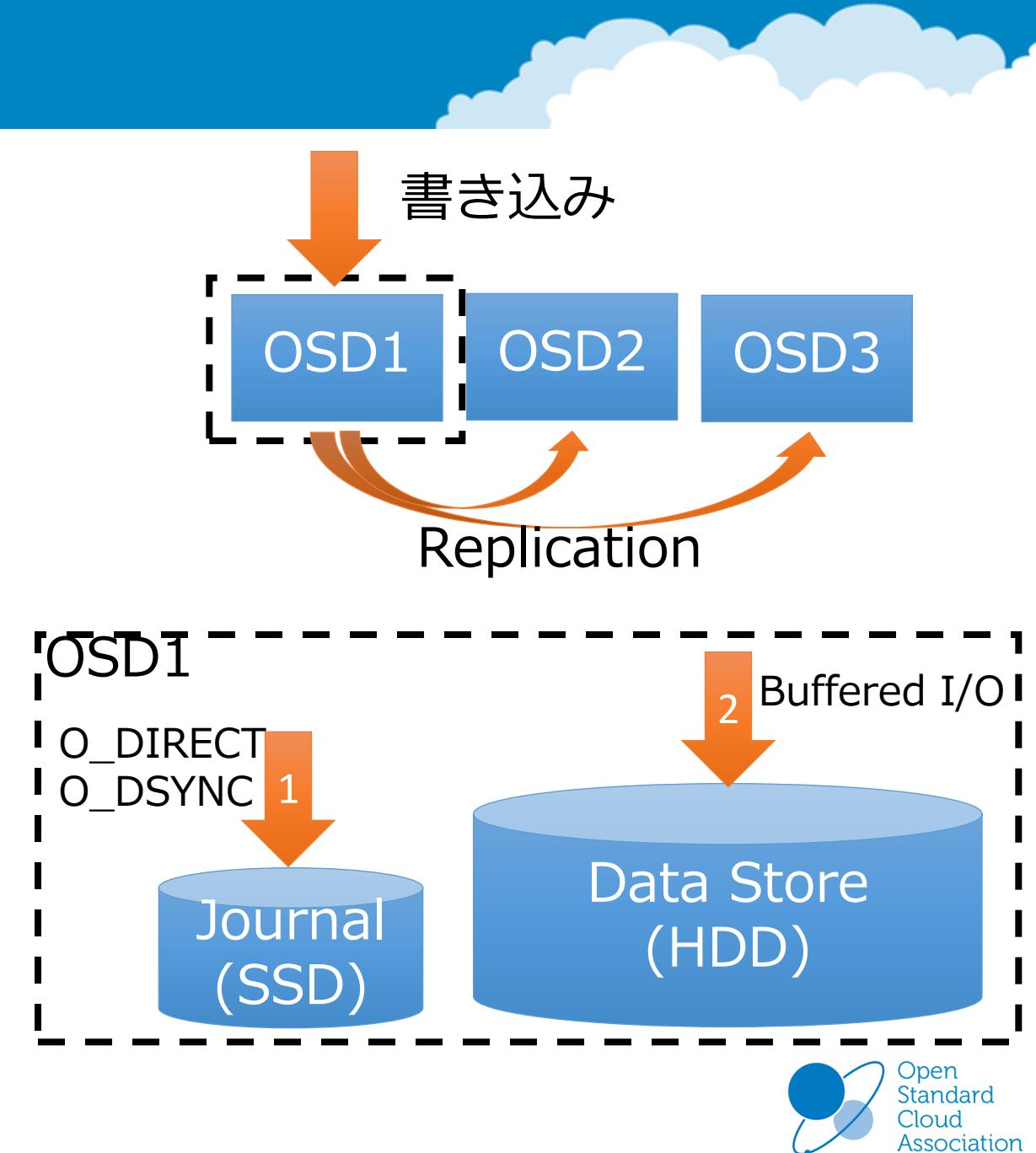


※ <http://www.openstack.org/assets/survey/April-2016-User-Survey-Report.pdf>

## 2.4 Journal & Replication

- データの永続性向上の為にOSD間でReplication  
(Erasure Codingも可能)
- Full Data Journalによる確実な書き込み
- JournalをSSD化することで、Write Cache的な効果
- 書き込み量が増えると、HDD側の書き込みがボトルネック

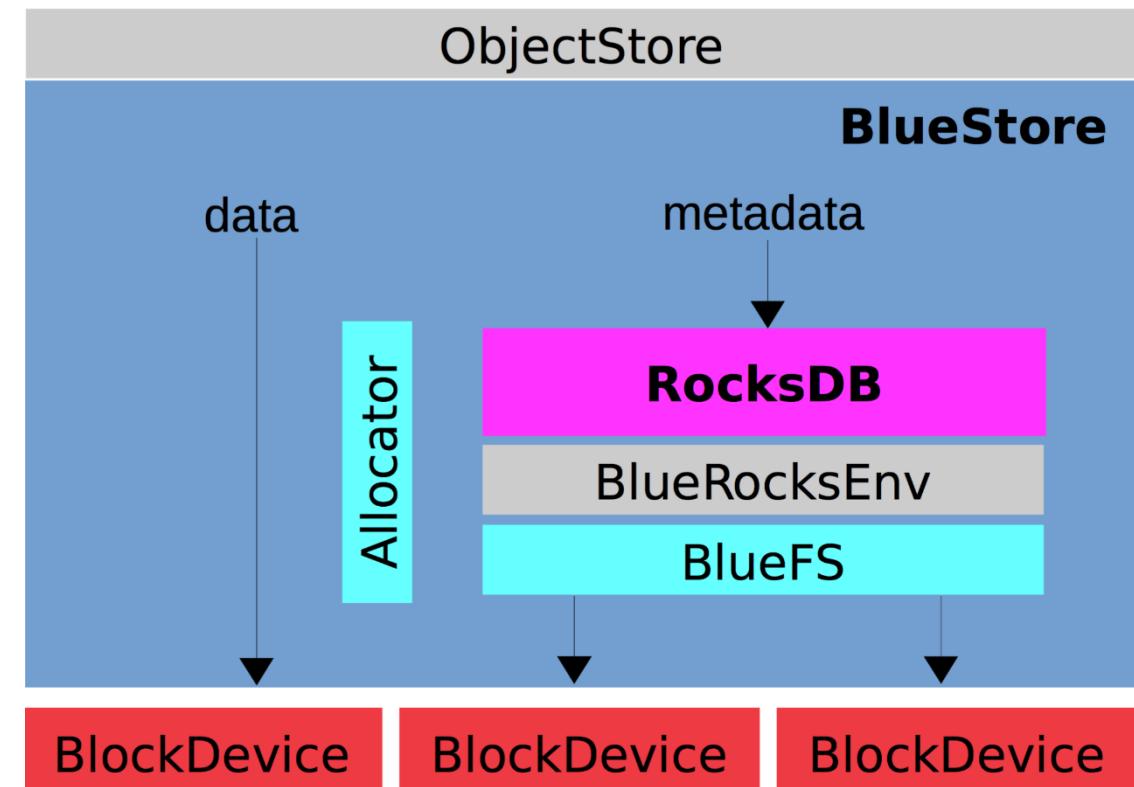
今回の検証は継続書き込みの為、HDD側の書き込みがボトルネック



## 2.5 BlueStoreによるCephの高速化

- ・新しいバックエンド **BlueStore**
- ・RHCS2 (**Jewel**)でリリース (Tech Preview)
- ・2~3倍の性能向上を期待
  - ・Full Data Journalが不要
  - ・Block Deviceへの直接書き込み

今回の検証はこれじゃないです  
念のため。。。



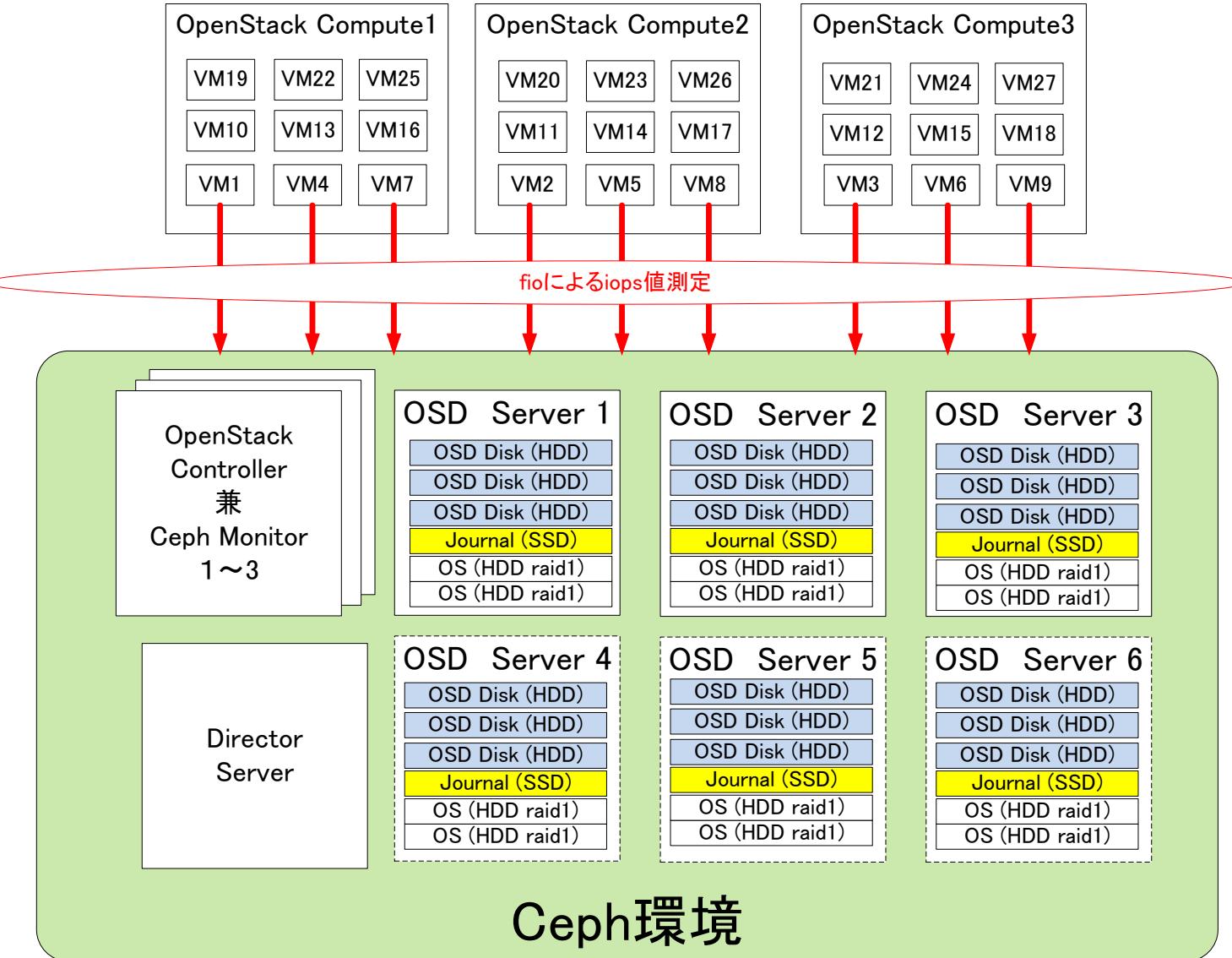
参考: <http://www.slideshare.net/sageweil1/bluestore-a-new-faster-storage-backend-for-ceph>  
<http://redhatstorage.redhat.com/2016/06/23/the-milestone-of-red-hat-ceph-storage-2/>

# 3. 検証方法及び結果

株式会社日立ソリューションズ  
技術開発本部 研究開発部  
技師  
工藤 雄大



# 3.1 ベンチマーク環境



## ・インスタンス(VM)

- 各Compute Serverに9VM作成(計27VM )
- Ceph RBDでBlock Deviceとしてマウント
- RHEL 7.2上に、fio-2.1.7をインストール
- Floating IPを割り当て
- Director ServerからパスワードレスSSHで、fioを実行可能に設定

## ・OSD Server(Svr)

- 6台準備。物理Disk仕様は下記。
  - OS : 300GB SAS 10Krpm x2(raid1)
  - OSD Disk : 600GB SAS 10Krpm x3 (JBOD)
  - Journal : 320GB SSD

(各OSD用に50GBのVolumeを準備)

## ・各OSD(Disk)へ下記パラメータ投入

```
# ceph tell osd.[OSD No] injectargs --journal_max_write_entries 1000  
--journal_max_write_bytes 1048576000 --journal_queue_max_ops  
3000 --journal_queue_max_bytes 1048576000 --  
filestore_max_sync_interval 10
```

## 3.2 ベンチマーク方法

Director ServerからVMへSSH接続し、VM上でfioを実行

```
ssh (user@instance IP) fio -rw=[read/write] -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=120 -bs=[BlockSize]k -numjobs=[Job数] -group_reporting > (file name) & ssh.....
```

- ・下記を変えながら2分ベンチマーク実行、2分sleep

- VM:1|2|3|9|27
- [read/write]:randread|randwrite
- [BlockSize]:4|16|32|64|128
- [Job数]:1|4|8|16
- 3回繰り返し
- OSD Svr :3|4|5|6 台

$$(1+2+3+9+27)$$

× 2

× 5

× 4

× 3

× 4

=20,160



更に失敗データ  
(後述)が5,000件程

### 3.3 (余談)ベンチマーク用スクリプト

```
62|
63echo "***R_1G_${i}KB_job${k}_para3_loop${i}***"
64ssh cloud-user@${c1} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para3_loop${i}_${c1}.log" & ssh cloud-user@${c2} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para3_loop${i}_${c2}.log" & ssh cloud-user@${c3} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para3_loop${i}_${c3}.log" &
65sleep ${s}s
66|
67echo "***R_1G_${i}KB_job${k}_para3_loop${i}***"
68ssh cloud-user@${c1} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para3_loop${i}_${c1}.log" & ssh cloud-user@${c2} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para3_loop${i}_${c2}.log" & ssh cloud-user@${c3} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para3_loop${i}_${c3}.log" &
69sleep ${s}s
70|
71echo "***R_1G_${i}KB_job${k}_para27_loop${i}***"
72ssh cloud-user@${c1} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para27_loop${i}_${c1}.log" & ssh cloud-user@${c2} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para27_loop${i}_${c2}.log" & ssh cloud-user@${c3} fio -rw=randread -size=1G -ioengine=libaio -iodepth=4 -invalidate=1 -direct=1 -name=test.bin -runtime=${t} -bs=${j}k -numJobs=${k} -group_reporting > "ReOSD6_R_1G_${i}KB_job${k}_para27_loop${i}_${c3}.log" &
73sleep ${s}s
74|
75done|
76done|
77done|
```

スクリプト完了まで  
30時間！！！

( × OSD Svrの台数変えて4回)

## 3.4 データ整形

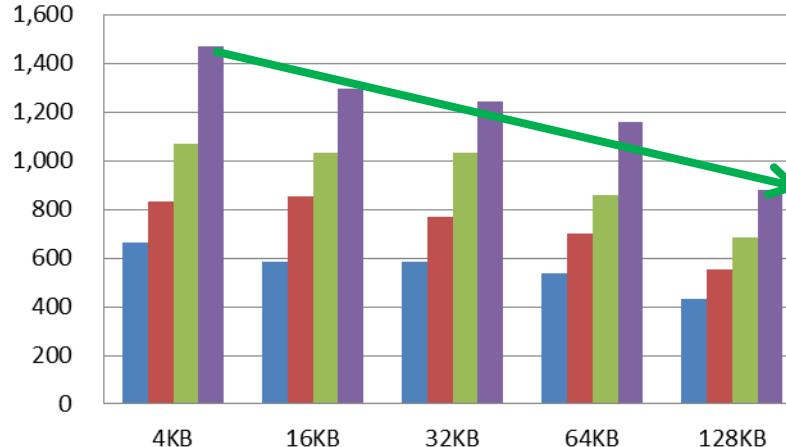
20,160のfioデータからIOPS、BWを抽出 => VM並列実行時のものを総和  
=> 3回繰り返しの中間値を採用 => 整形(レコード数800)

WhitePaperの180以上のグラフから、  
BlockSize(BS)、VM数、OSD Svr数観点で抜粋してご説明します。

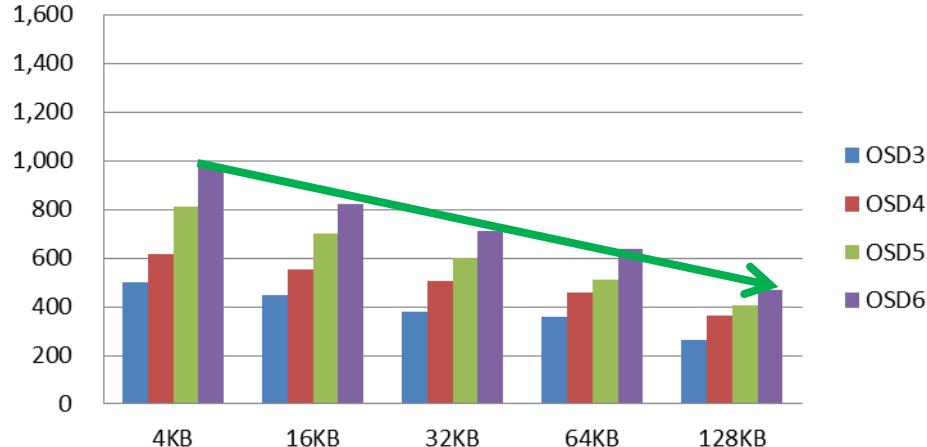
| 1  | ノード数 | r/w | bs | job数 | 並列度   | iops中央 | w単位変 | iops総和 | bw総和    |            |
|----|------|-----|----|------|-------|--------|------|--------|---------|------------|
| 2  | OSD3 | R   | 1G | 4KB  | job1  | para1  | 3309 | 32798  | 3,309   | 32,798     |
| 3  | OSD3 | R   | 1G | 4KB  | job1  | para2  | 3099 | 30999  | 8,086   | 101,882    |
| 4  | OSD3 | R   | 1G | 4KB  | job1  | para3  | 2805 | 29440  | 11,980  | 205,840    |
| 5  | OSD3 | R   | 1G | 4KB  | job1  | para9  | 1944 | 28046  | 24,205  | 1,418,512  |
| 6  | OSD3 | R   | 1G | 4KB  | job1  | para27 | 1079 | 26751  | 67,652  | 6,223,472  |
| 7  | OSD3 | R   | 1G | 4KB  | job4  | para1  | 4880 | 89201  | 4,880   | 89,201     |
| 8  | OSD3 | R   | 1G | 4KB  | job4  | para2  | 4164 | 85234  | 16,495  | 294,030    |
| 9  | OSD3 | R   | 1G | 4KB  | job4  | para3  | 3391 | 83275  | 22,413  | 577,475    |
| 10 | OSD3 | R   | 1G | 4KB  | job4  | para9  | 1443 | 78766  | 42,596  | 3,062,667  |
| 11 | OSD3 | R   | 1G | 4KB  | job4  | para27 | 586  | 71329  | 121,194 | 11,762,322 |
| 12 | OSD3 | R   | 1G | 4KB  | job8  | para1  | 5666 | 144323 | 5,666   | 144,323    |
| 13 | OSD3 | R   | 1G | 4KB  | job8  | para2  | 3959 | 138380 | 20,206  | 500,911    |
| 14 | OSD3 | R   | 1G | 4KB  | job8  | para3  | 3352 | 130625 | 28,960  | 900,435    |
| 15 | OSD3 | R   | 1G | 4KB  | job8  | para9  | 1288 | 123735 | 71,398  | 4,571,412  |
| 16 | OSD3 | R   | 1G | 4KB  | job8  | para27 | 557  | 110660 | 215,919 | 17,812,896 |
| 17 | OSD3 | R   | 1G | 4KB  | job16 | para1  | 8735 | 240806 | 8,735   | 240,806    |
| 18 | OSD3 | R   | 1G | 4KB  | job16 | para2  | 5900 | 224640 | 33,961  | 811,584    |
| 19 | OSD3 | R   | 1G | 4KB  | job16 | para3  | 3962 | 229206 | 51,929  | 1,465,848  |
| 20 | OSD3 | R   | 1G | 4KB  | job16 | para9  | 1649 | 205309 | 160,891 | 6,819,710  |
| 21 | OSD3 | R   | 1G | 4KB  | job16 | para27 | 556  | 178178 | 494,491 | 23,711,297 |

### 3.5 BS増加時の傾向(Write : VM1,27)

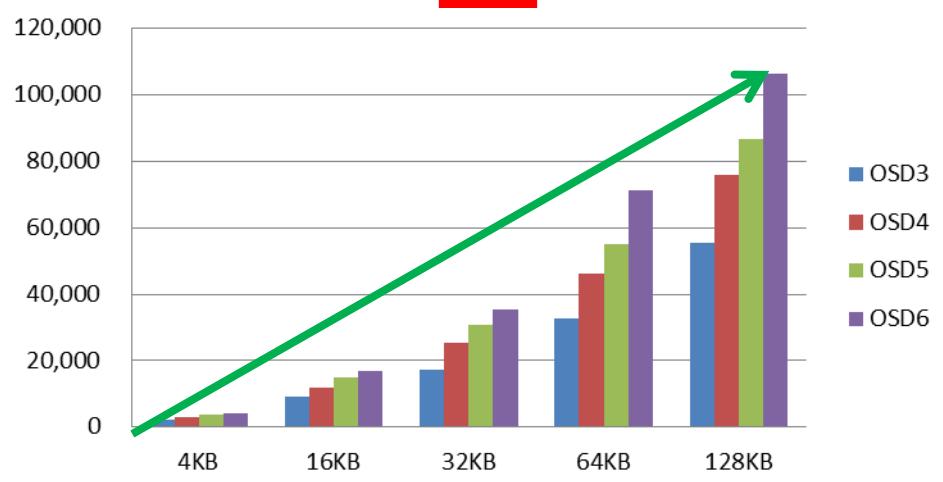
IOPS\_W : VM1 job16 (io/s)



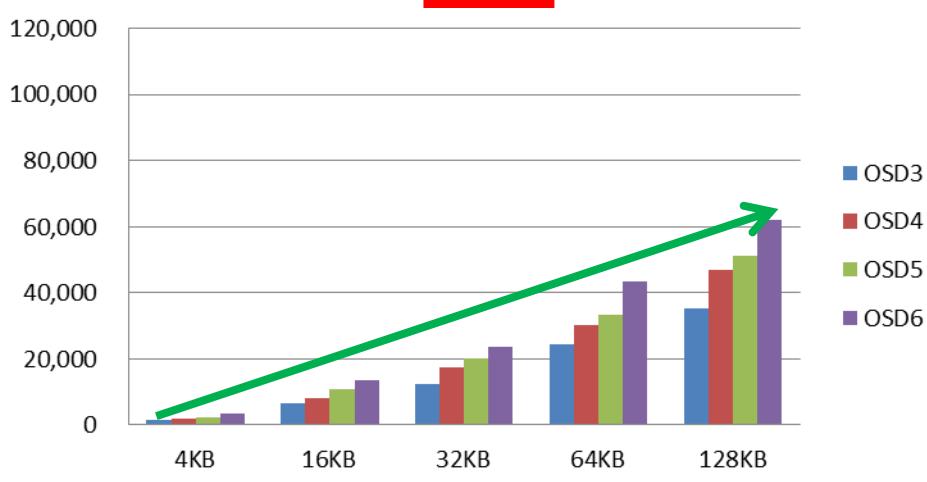
IOPS\_W : VM27 job16 (io/s)



Throughput\_W : VM1 job16 (KB/s)



Throughput\_W : VM27 job16 (KB/s)



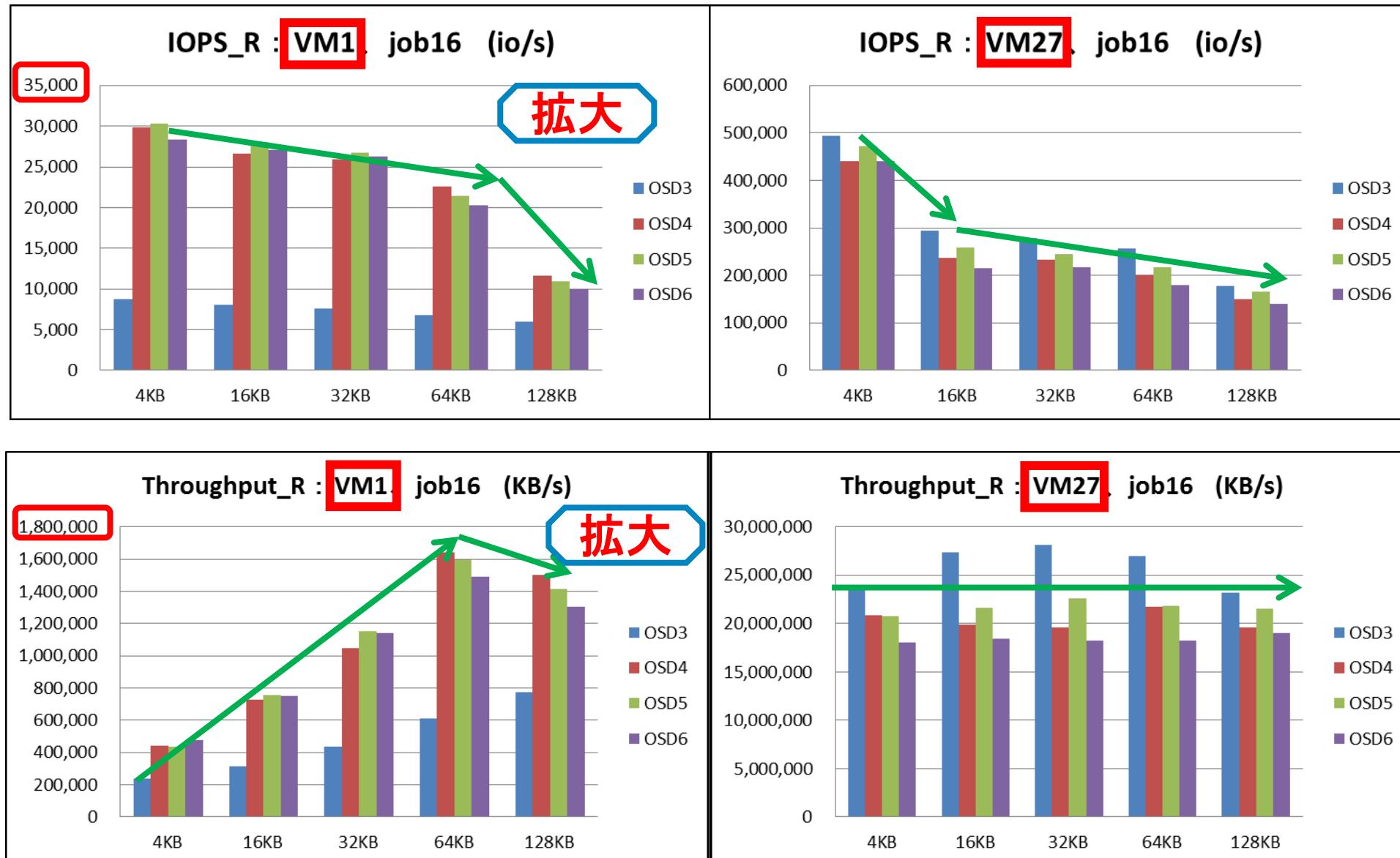
#### IOPS

- ✓ VM1台時  
128KB時は、4KB時の0.6倍の性能
- ✓ VM27台時  
128KB時は、4KB時の0.5倍の性能

#### Throughput

- ✓ VM1台時  
128KB時は、4KB時の25倍の性能
- ✓ VM27台時  
128KB時は、4KB時の17倍の性能

### 3.6 BS増加時の傾向(Read : VM1,27)



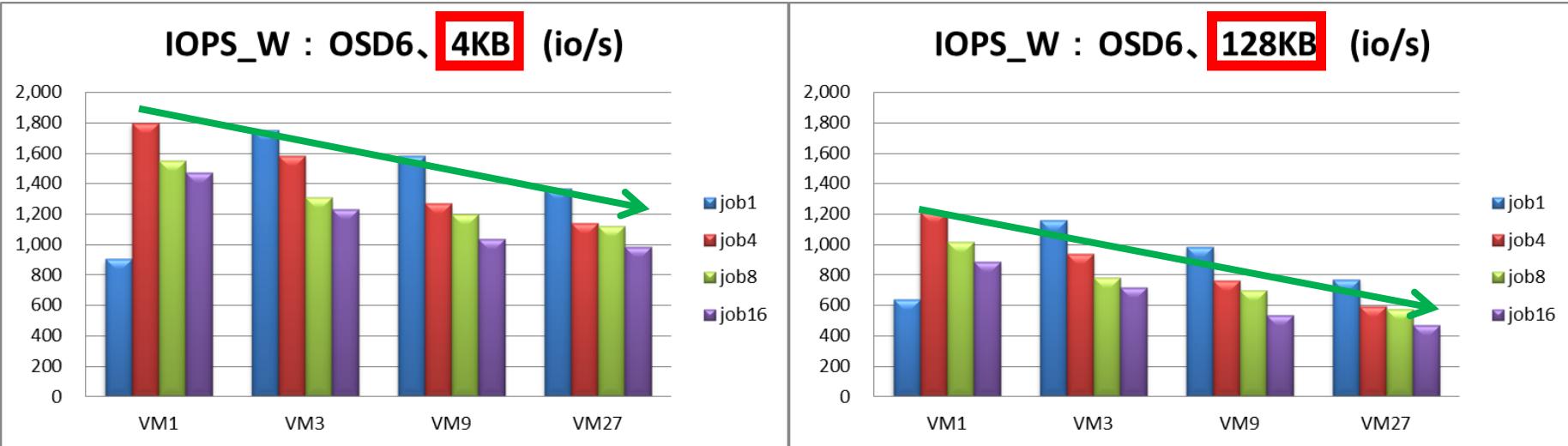
#### IOPS

- ✓ VM1台時  
128KB時は、4KB時の0.4倍の性能(OSD3台時は低い)
- ✓ VM27台時  
128KB時は、4KB時の0.3倍の性能

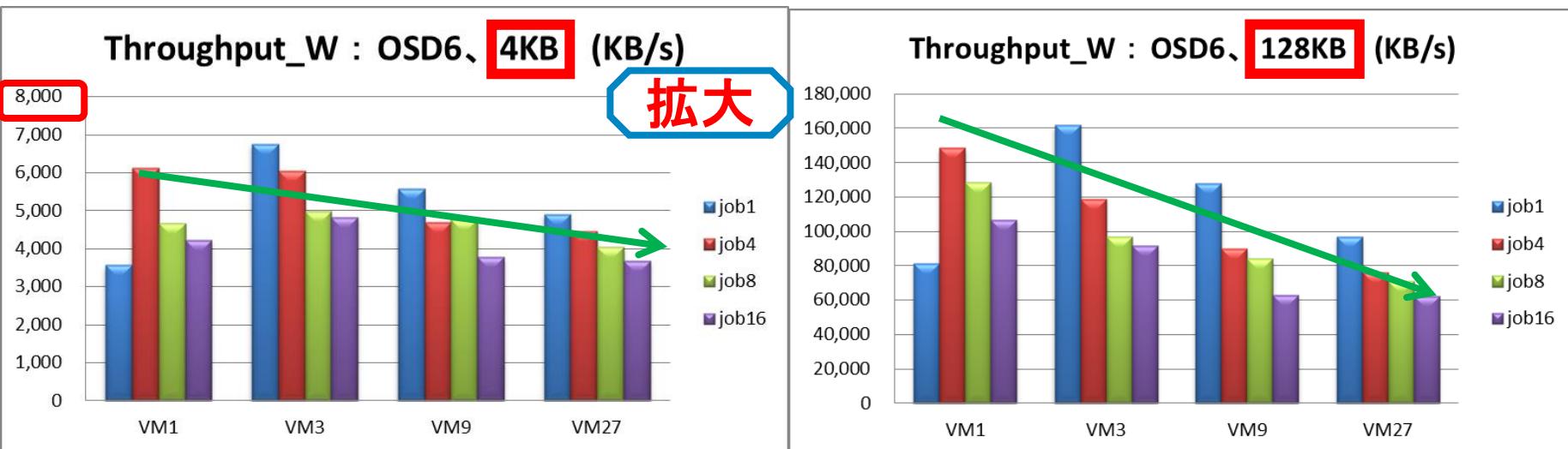
#### Throughput

- ✓ VM1台時  
128KB時は、4KB時の2.7倍の性能(最大は64KB時)
- ✓ VM27台時  
BSにより性能は変わらず

### 3.7 VM数増加時の傾向(Write : BS 4KB,128KB)



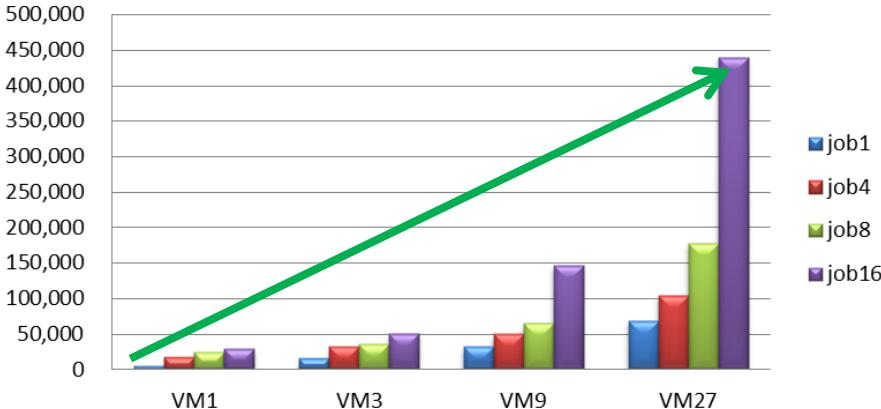
- **IOPS**
- ✓ 4KB時  
VM27台時は、VM1台時の0.7倍の性能
- ✓ 128KB時  
VM27台時は、VM1台時の0.5倍の性能  
(共にjob1時除く)



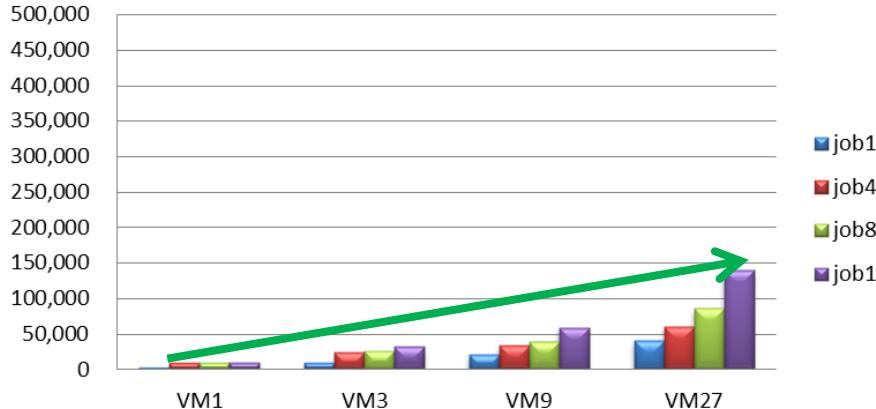
- **Throughput**
- ✓ 4KB時  
VM27台時は、VM1台時の0.8倍の性能
- ✓ 128KB時  
VM27台時は、VM1台時の0.6倍の性能  
(共にjob1時除く)

### 3.8 VM数増加時の傾向(Read : BS 4KB,128KB)

IOPS\_R : OSD6、**4KB** (io/s)



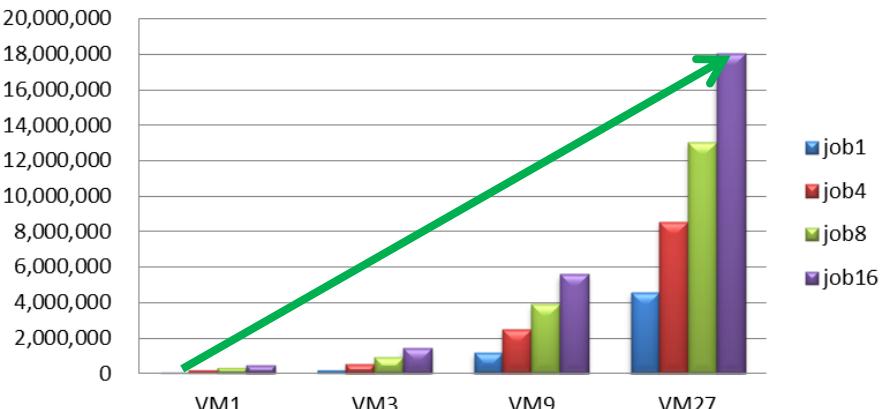
IOPS\_R : OSD6、**128KB** (io/s)



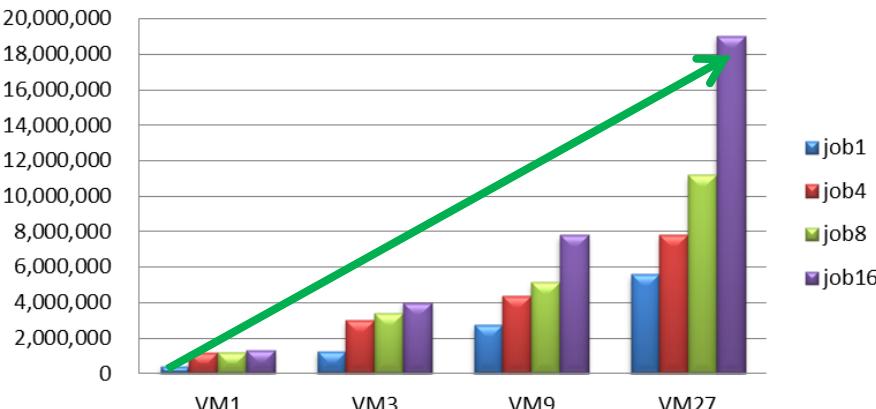
#### IOPS

- ✓ 4KB時  
VM27台時は、VM1台時の16倍の性能
- ✓ 128KB時  
VM27台時は、VM1台時の14倍の性能

Throughput\_R : OSD6、**4KB** (KB/s)



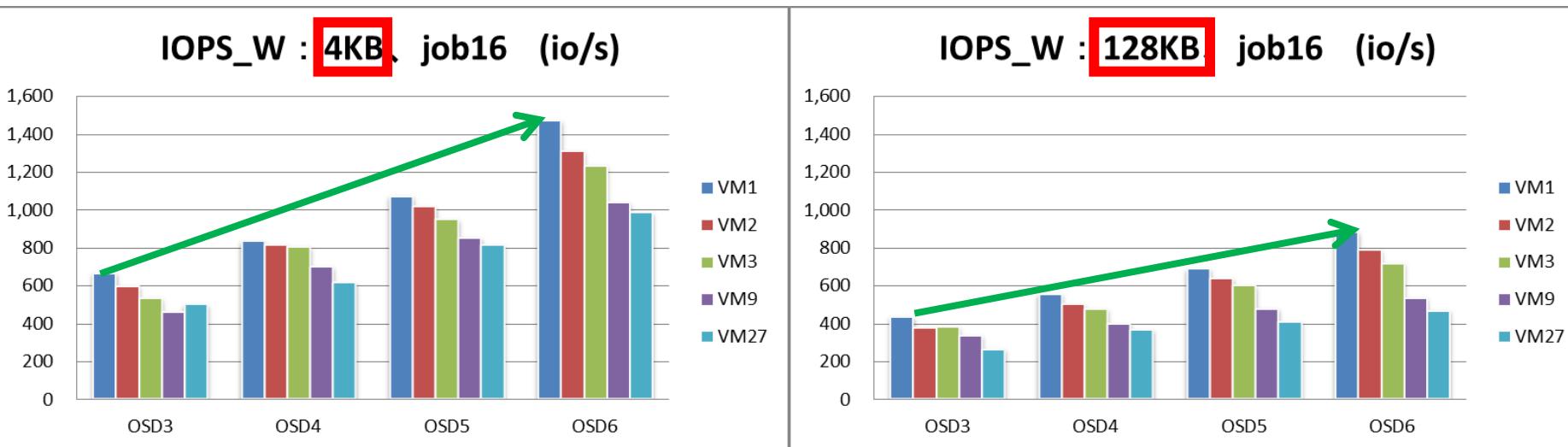
Throughput\_R : OSD6、**128KB** (KB/s)



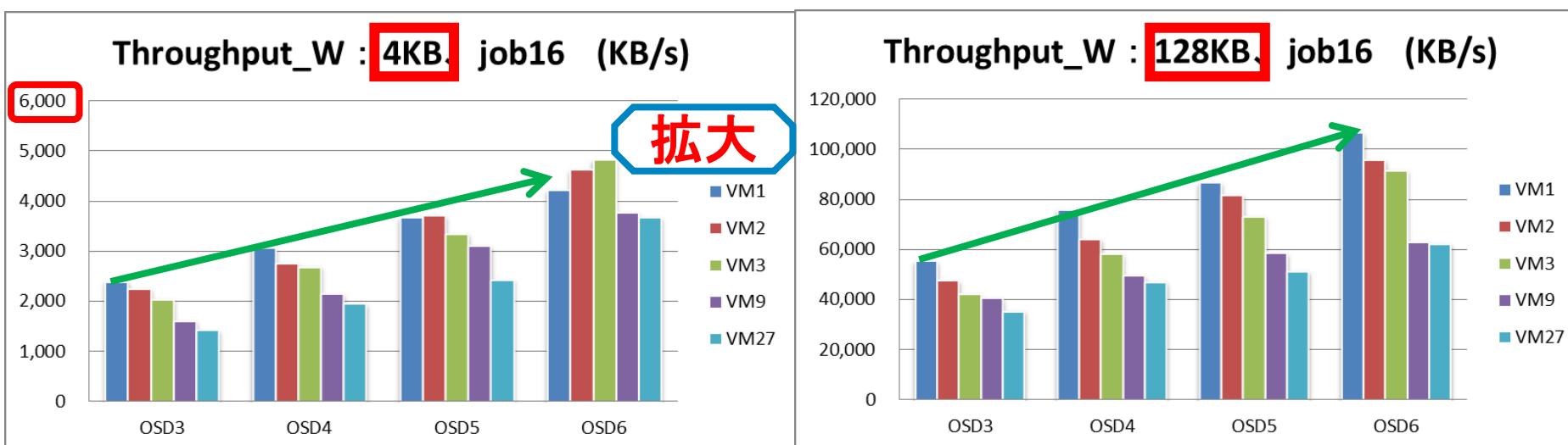
#### Throughput

- ✓ 4KB時  
VM27台時は、VM1台時の38倍の性能
- ✓ 128KB時  
VM27台時は、VM1台時の14倍の性能

### 3.9 OSD Svr增加時の傾向(Write : BS 4KB,128KB)

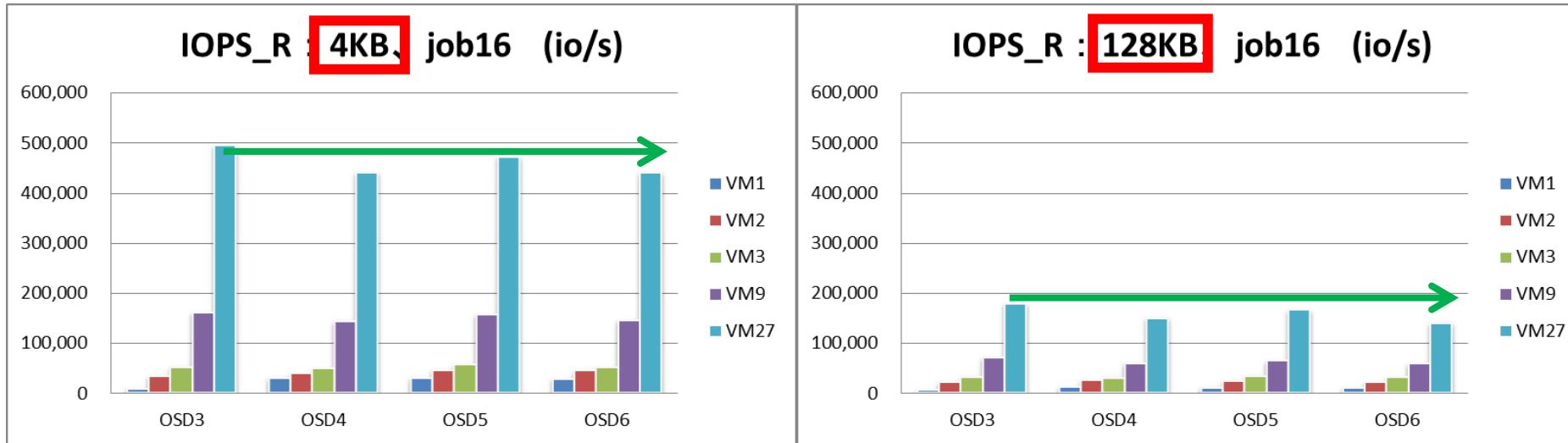


- **IOPS**
  - ✓ OSD Svr台数増加に伴い性能向上

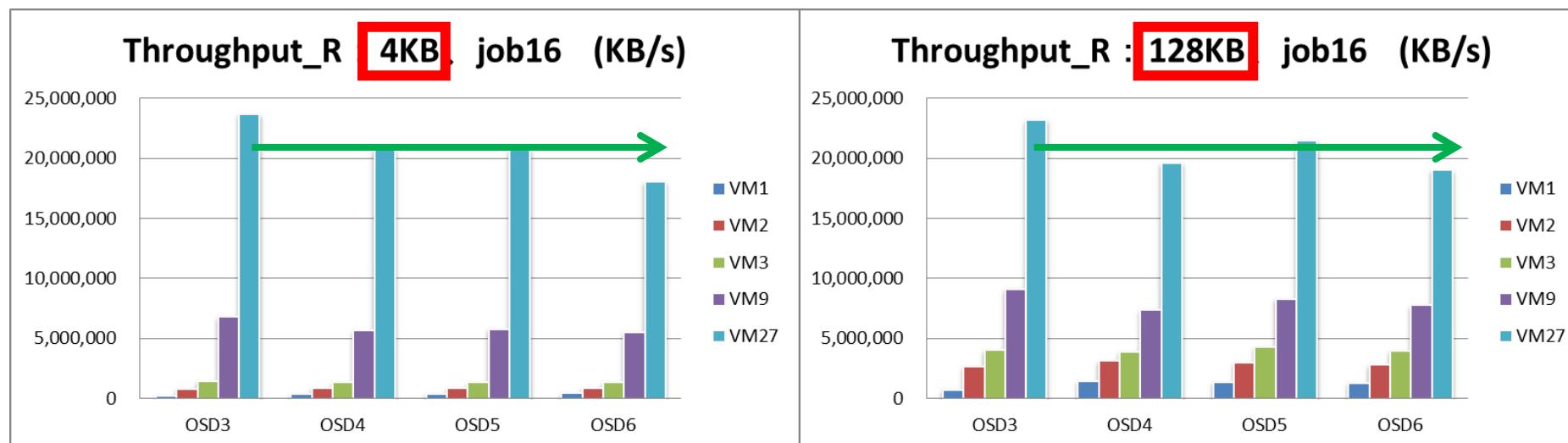


- **Throughput**
  - ✓ OSD Svr台数増加に伴い性能向上

### 3.10 OSD Svr増加時の傾向(Read : BS 4KB,128KB)



- IOPS
  - ✓ OSD Svr台数増加してもあまり変化無し



- Throughput
  - ✓ OSD Svr台数増加してもあまり変化無し(3台時が少し高い)

### 3.11 全体傾向

- BlockSizeからみた全体傾向

| # | パターン | Random Write |            | Random Read |            |
|---|------|--------------|------------|-------------|------------|
|   |      | IOPS         | Throughput | IOPS        | Throughput |
| 1 | BS増加 | 緩やかに低下       | 大幅に向       | 緩やかに低下      | 大幅に向       |

普通のDiskストレージと同じ傾向

- OSD Svr数からみた全体傾向

| # | パターン       | Random Write |            | Random Read |            |
|---|------------|--------------|------------|-------------|------------|
|   |            | IOPS         | Throughput | IOPS        | Throughput |
| 1 | OSD Svr数増加 |              | 比例的に向上     |             | 変化なし       |

WriteはOSD Server数増加に伴い向上  
Readは限界性能までベンチをかけきれず

### 3.12 最大性能値

- ・今回の検証で得られた全データにおける、最大値

| # | 項目            | 条件                                      | 値(io/s) |
|---|---------------|---|---------|
| 1 | Write時 最大IOPS | OSD Svr:6台、Block Size:16KB、Job数:4、VM:1  | 1,805   |
| 2 | Read時 最大IOPS  | OSD Svr:3台、Block Size:4KB、Job数:16、VM:27 | 494,491 |

| # | 項目                  | 条件                                       | 値(KB/s)    |
|---|---------------------|--|------------|
| 3 | Write時 最大Throughput | OSD Svr:6台、Block Size:128KB、Job数:1、VM:3  | 161,525    |
| 4 | Read時 最大Throughput  | OSD Svr:3台、Block Size:32KB、Job数:16、VM:27 | 28,111,873 |

✓ 今回の検証の範囲(連続的な負荷をかけた場合)において

- JournalのWrite Cacheはあまり効果無し  
(性能はOSD Diskの本数に依存)
- JournalのRead Cacheは絶大な効果
  - 本検証ではOSD Svr 3台時のRead限界にすら達せず
  - OpenStack Compute側でのメモリキャッシュ可能性

# 4. TIPS & 検証まとめ

株式会社日立ソリューションズ  
技術開発本部 研究開発部  
平原 一帆



# Ceph性能測定TIPS ~と言う名の検証失敗談

- 
- 4.1 Ceph DeployによるCeph + OpenStack連携かなり大変だよ問題
  - 4.2 Diskサイズを揃えないと性能測定の結果がバラつくよ問題
  - 4.3 HDD台数が少ないからWrite速度遅いんじゃないの問題
  - 4.4 まとめ

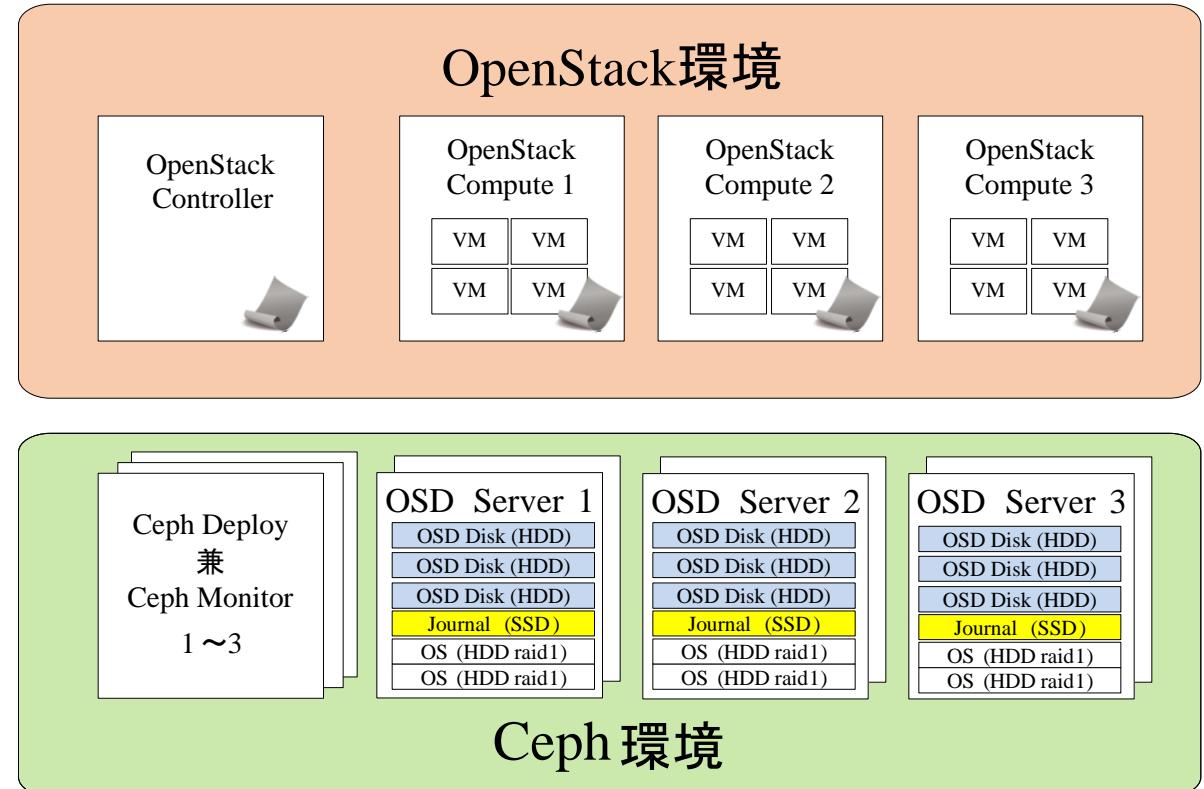
本章は検証失敗談に基づく『考察』のため、推測が含まれています！

## 4.1 Ceph DeployによるCeph+OpenStack連携問題(1/4)

Ceph DeployによるCeph + OpenStack連携かなり大変だよ問題

### ■OpenStack+Ceph環境構築

- 1.Ceph Deploy環境つくる
- 2.Ceph OSD環境つくる
- 3.Ceph DeployでOSD、Monitor 設定
- 4.block volume作成、配置、マウント
- 5.**OpenStack Controllerつくる**
- 6.**OpenStack Computeつくる**
- 7.**OpenStack Flavorの変更**
- 8.**OpenStackの関連設定ファイルの変更**



# 4.1 Ceph DeployによるCeph+OpenStack連携問題(2/4)

The screenshot shows the 'INSTALLATION (QUICK)' section of the Ceph documentation. It includes a 'TABLE OF CONTENTS' with links to various Ceph components and a 'QUICK SEARCH' bar. The main content is 'STEP 1: PREFLIGHT', which details the preflight checklist required before deploying a Ceph Storage Cluster. It lists several steps under 'Preflight' and 'Ceph Node Setup'.

## INSTALLATION (QUICK)

### STEP 1: PREFLIGHT

A *Ceph Client* and a *Ceph Node* may require some basic configuration work prior to deploying a Ceph Storage Cluster. You can also avail yourself of help by getting involved in the Ceph community.

- **Preflight**
  - **Ceph Deploy Setup**
    - Advanced Package Tool (APT)
    - Red Hat Package Manager (RPM)
  - **Ceph Node Setup**
    - Install NTP
    - Install SSH Server
    - Create a Ceph Deploy User
    - Enable Password-less SSH
    - Enable Networking On Bootup
    - Ensure Connectivity
    - Open Required Ports
    - TTY
    - SELinux
    - Priorities/Preferences
  - **Summary**

## Ceph構築に使ったコマンド ([docs.ceph.com/docs/](https://docs.ceph.com/docs/)より抜粋)

```
sudo subscription-manager repos --enable=rhel-7-server-extras-rpms
sudo yum install -y yum-utils && sudo yum-config-manager --add-repo
https://dl.fedoraproject.org/pub/epel/7/x86_64/ && sudo yum install --
nogpgcheck -y epel-release && sudo rpm --import /etc/pki/rpm-gpg/RPM-
GPG-KEY-EPEL-7 && sudo rm /etc/yum.repos.d/dl.fedoraproject.org*
sudo vim /etc/yum.repos.d/ceph.repo
sudo yum update && sudo yum install ceph-deploy
sudo yum install ntp ntpdate ntp-doc
ssh user@ceph-server
sudo useradd -d /home/{username} -m {username}
sudo passwd {username}
echo "{username} ALL=(root) NOPASSWD:ALL" | sudo tee
/etc/sudoers.d/{username}
sudo chmod 0440 /etc/sudoers.d/{username}
ssh-keygen
ssh-copy-id {username}@node1
sudo firewall-cmd --zone=public --add-port=6789/tcp --permanent
sudo iptables -A INPUT -i {iface} -p tcp -s {ip-address}/{netmask} --dport
6789 -j ACCEPT
sudo setenforce 0
mkdir my-cluster
cd my-cluster
ceph-deploy purgedata {ceph-node} [{ceph-node}]
ceph-deploy forgetkeys
```

```
ceph-deploy purge {ceph-node} [{ceph-node}]
ceph-deploy new {initial-monitor-node(s)} ceph-deploy new node1
ceph-deploy install {ceph-node} [{ceph-node} ...]
ceph-deploy install admin-node node1 node2 node3
ceph-deploy mon create-initial
ceph-deploy osd prepare {ceph-node}:/path/to/directory
ceph-deploy osd activate {ceph-node}:/path/to/directory
ceph-deploy admin {admin-node} {ceph-node}
sudo chmod +r /etc/ceph/ceph.client.admin.keyring
ceph health
ceph-deploy osd prepare {ceph-node}:/path/to/directory
ceph-deploy osd activate osdserver1:/dev/sdb1:/dev/ssd1
ceph -w
ceph-deploy mon add {ceph-node}
ceph quorum_status --format json-pretty
rbd create foo --size 4096 [-m {mon-IP}] [-k
/path/to/ceph.client.admin.keyring]
sudo rbd map foo --name client.admin [-m {mon-IP}] [-k
/path/to/ceph.client.admin.keyring]
sudo mkfs.ext4 -m0 /dev/rbd/rbd/foo
sudo mkdir /mnt/ceph-block-device
sudo mount /dev/rbd/rbd/foo /mnt/ceph-block-device
cd /mnt/ceph-block-device
```

別途OpenStack構築が必要な上に、連携設定も必要...。

コマンドが多過ぎて間違いやすい上に、  
間違いに気付いてからのリカバリが複雑！

## 4.1 Ceph DeployによるCeph+OpenStack連携問題(3/4)

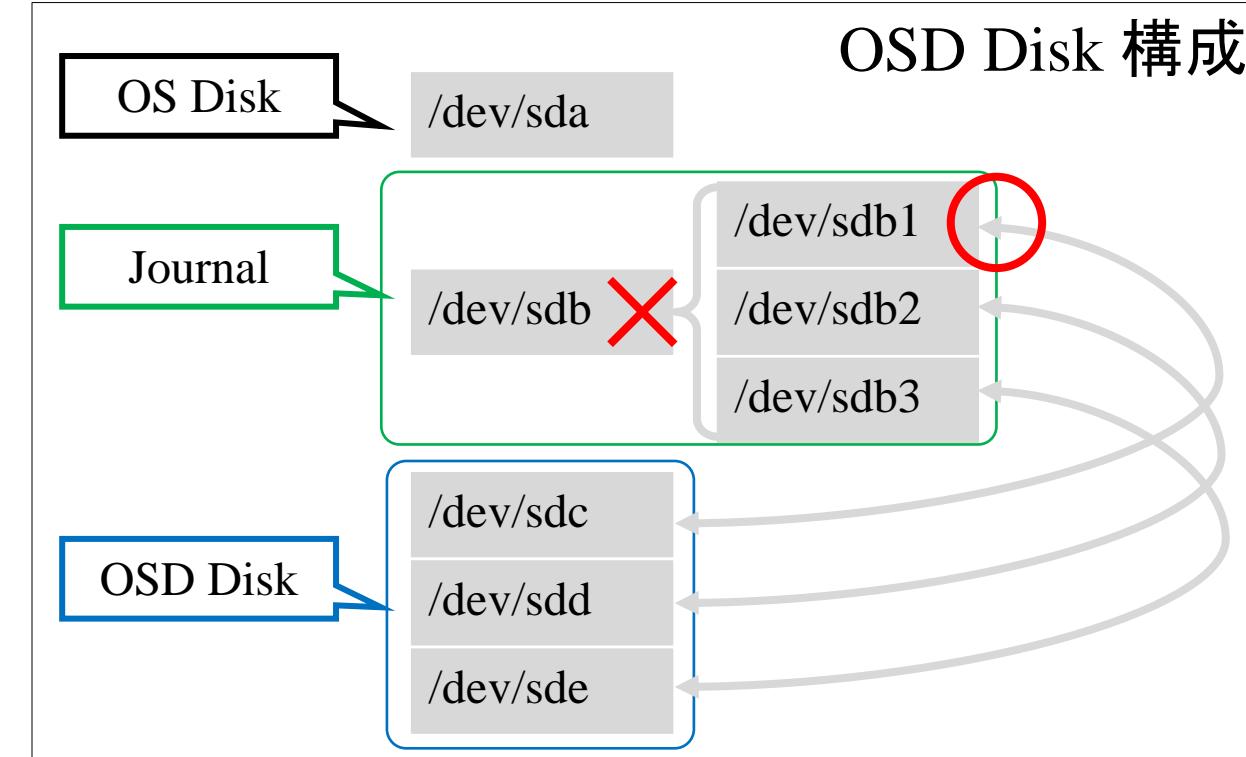
### ■実際にやった失敗の一例

正しいコマンド

```
ceph-deploy osd activate  
  \osdserver1:/dev/sdc:/dev/sdb1
```

間違えたコマンド

```
ceph-deploy osd activate  
  \osdserver1:/dev/sdc:/dev/sdb
```



→ エラーも特にせず、かつJournalが使用されず、性能測定結果に大きな影響が。

**細かすぎて、どこが間違っているかななか気付かない**

## 4.1 Ceph DeployによるCeph+OpenStack連携問題(4/4)

■要するに、(当初の私のように毛嫌いせずに)  
RHEL-OSP Directorのような構築ツールが  
あるならそれを利用しよう！

- ・Directorの管理下としてServerを登録
- ・Serverに役割を付与、Network設定

↓

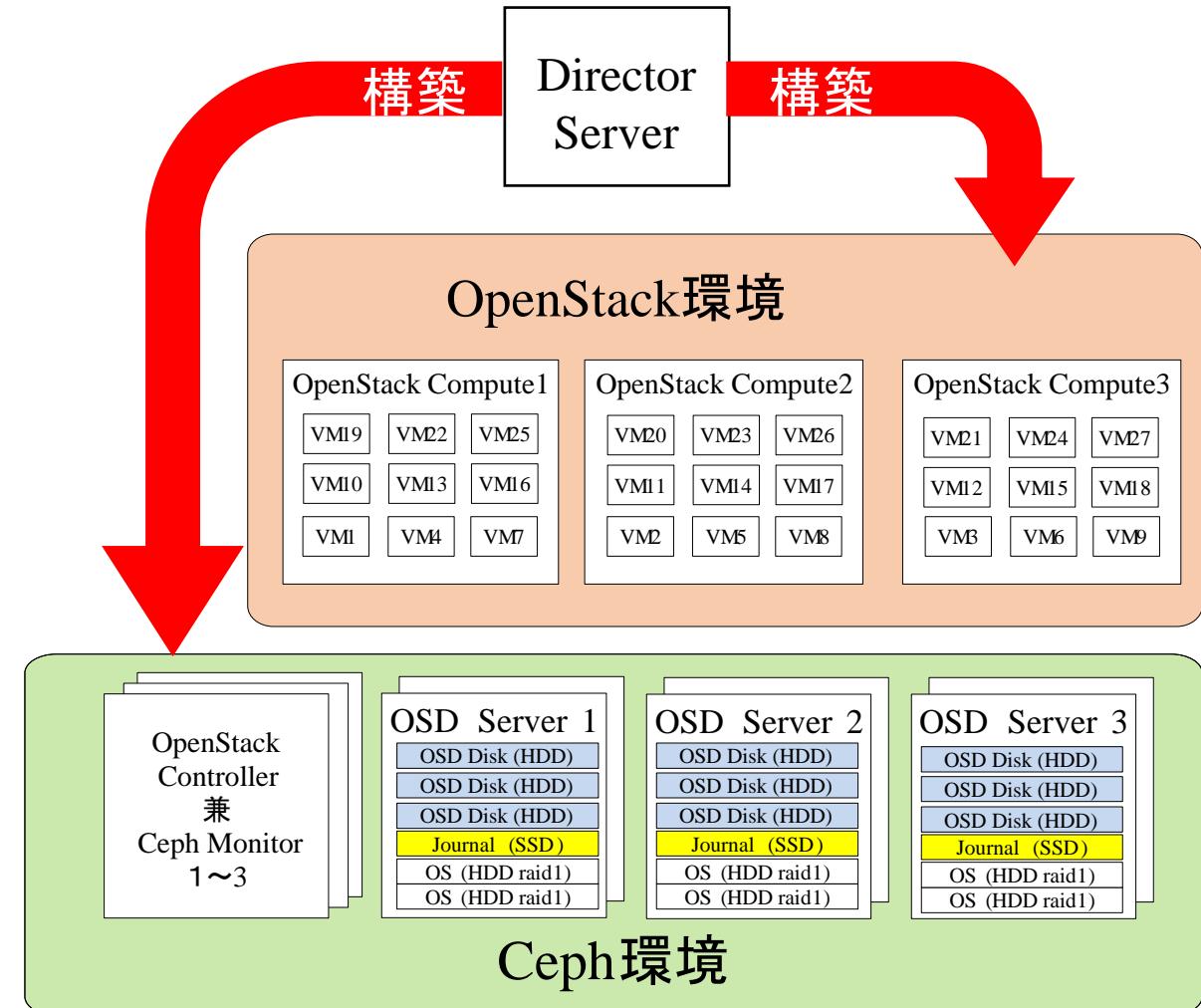
**設定をDirector Serverに集約、迅速化**

■というだけでなく、

- ・台数を変化させて再構築
- ・設定値を微妙に変更して再構築

↓

**再構築が簡単、放置して昼休憩可  
性能測定のようなScrap & Build向き**

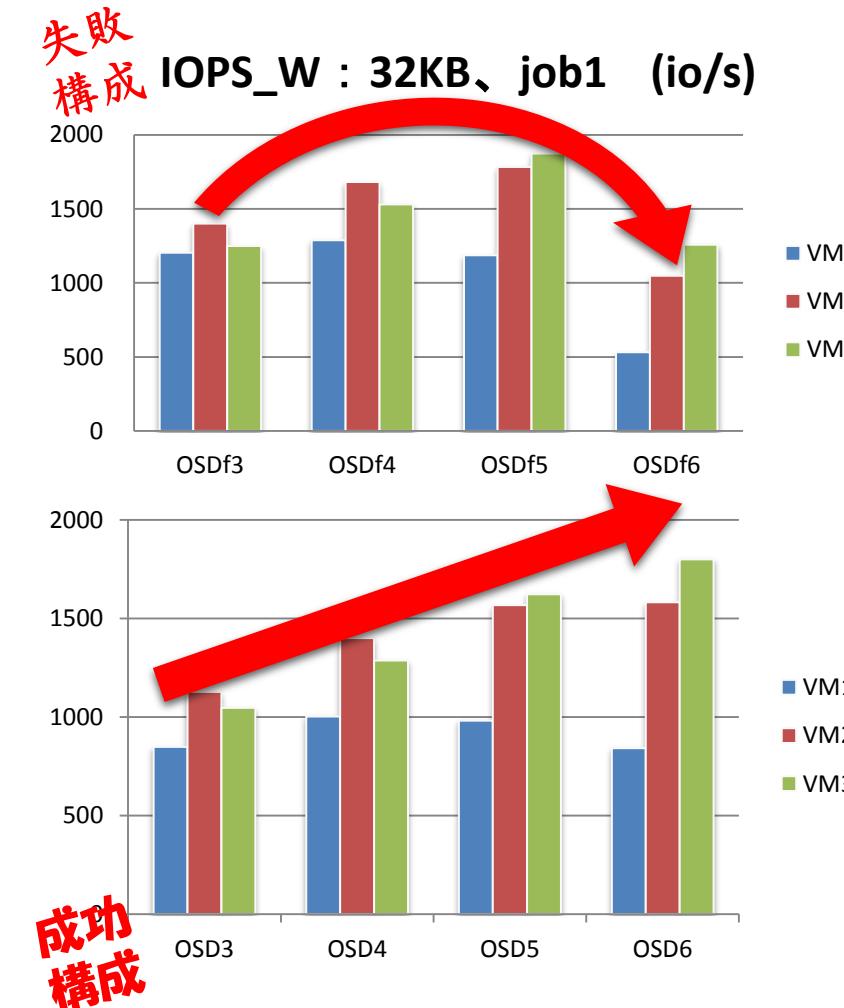


# Ceph性能測定TIPS

- 
- 4.1 Ceph DeployによるCeph + OpenStack連携かなり大変だよ問題  
→手入力がとても多いから可能な限りツールを使おうね
  - 4.2 Diskサイズを揃えないと性能測定の結果がバラつくよ問題
  - 4.3 HDD台数が少ないからWrite速度遅いんじゃないの問題
  - 4.4 まとめ

## 4.2 Disk揃えないと測定結果がバラつくよ問題(1/2)

Diskサイズを揃えないと性能測定の結果がバラつくよ問題



■当初想定していた性能検証結果

「OSD Serverが増えていけば、IOPS性能が向上していくだろう」

■初回性能測定

「OSD Server増やしたら性能すごく悪くなってる...？」  
→ どこかおかしいのでは？？

■色々見直す

→ OSD Serverに使ってるDiskサイズが問題なのかも？

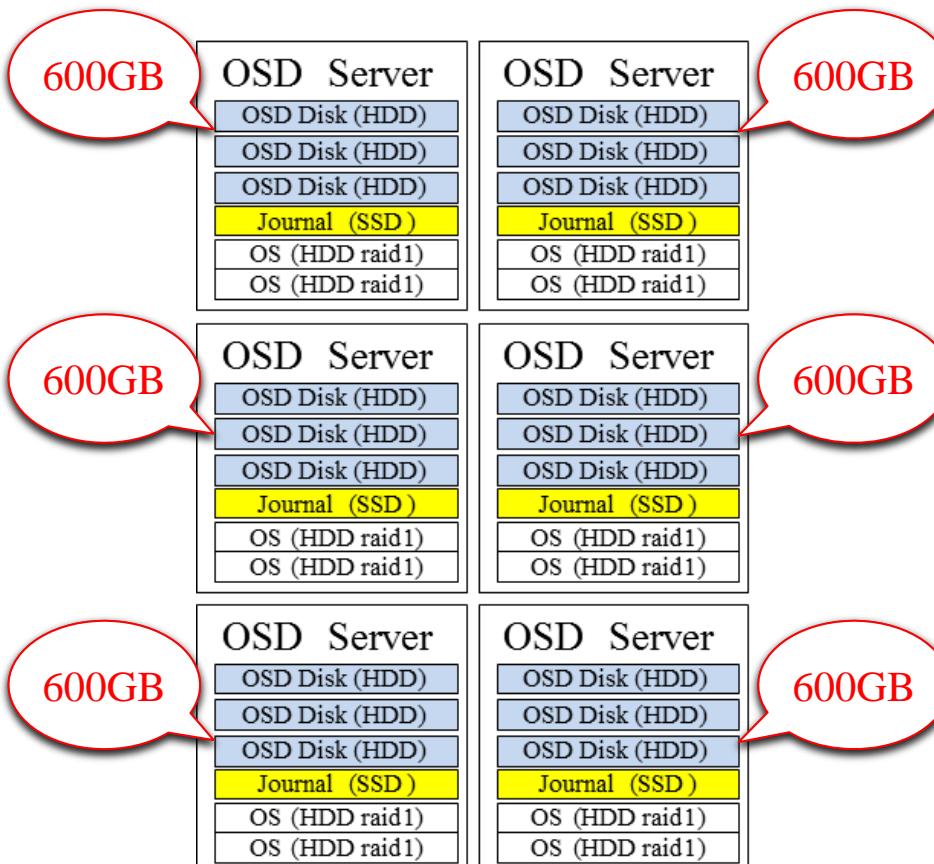
■構成見直し後、再度性能測定

だいたい想定どおりの挙動。

「OSD Serverを増やしていけば、性能が向上する！」

## 4.2 Disk揃えないと測定結果がバラつくよ問題(2/2)

→OSD Serverに使ってるDiskサイズの問題なのかも？



### ■ どういうこと？

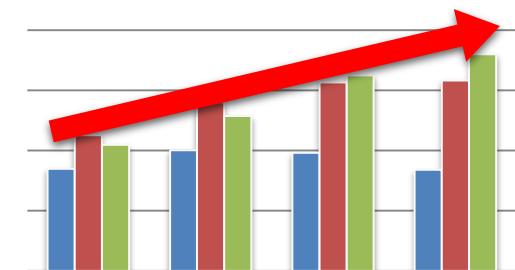
→OSD ServerのOSD(HDD)の中に300GBと600GBが混在

### ■ 全台600GBに統一したら想定通りの性能測定結果

→混在しているから性能がバラつくと予測

### ■ ここで疑問

→混在してるとなぜ性能がバラつくの？



→Diskの利用度合はweightで設定される。

ceph-deployコマンドやceph-diskコマンドによるDisk追加時、  
指定がない場合はDiskサイズに応じたweightが自動設定される。

→Diskサイズを揃えることで性能測定結果は安定

またはweightを揃えることでも安定すると推測

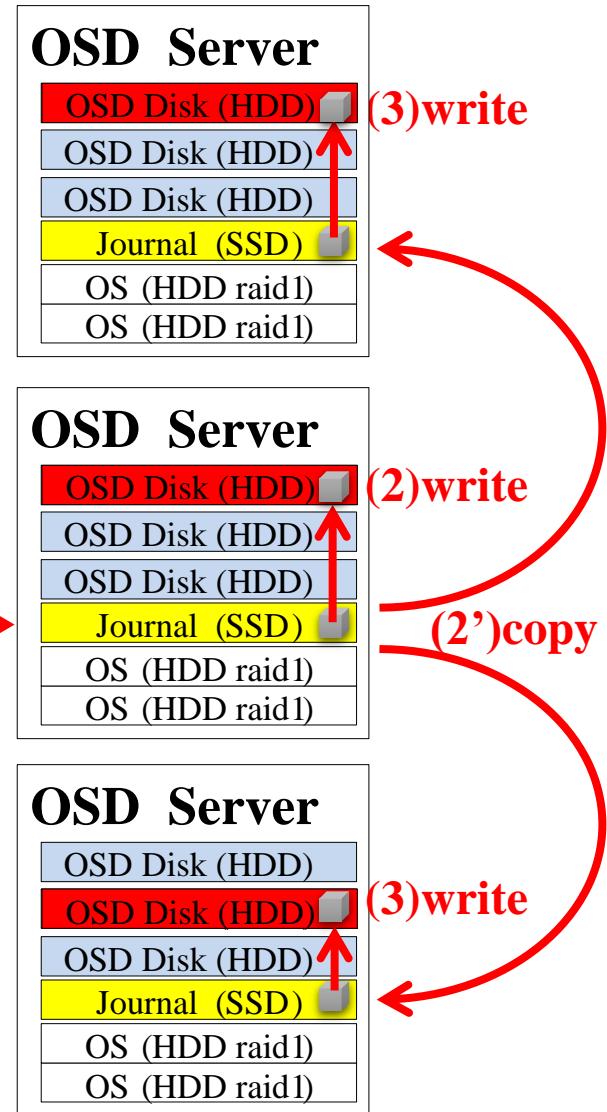
# Ceph性能測定TIPS

- 
- 4.1 Ceph DeployによるCeph + OpenStack連携かなり大変だよ問題  
→手入力がとても多いから可能な限りツールを使おうね
  - 4.2 Diskサイズを揃えないと性能測定の結果がバラつくよ問題  
→性能の安定を重視するなら同じDiskを選ぼうね
  - 4.3 HDD台数が少ないからWrite速度遅いんじゃないの問題
  - 4.4 まとめ

## 4.3 HDDが少ないからWrite速度遅いんじゃないの問題(1/4)



4KB



### (前章再掲)

✓ 今回の検証の範囲において  
JournalのWrite Cacheはあまり効果無し  
性能はOSD Diskの本数に依存

### ■ 理由はCephのアーキテクチャにあると推測

Cephではデータが渡された際に、

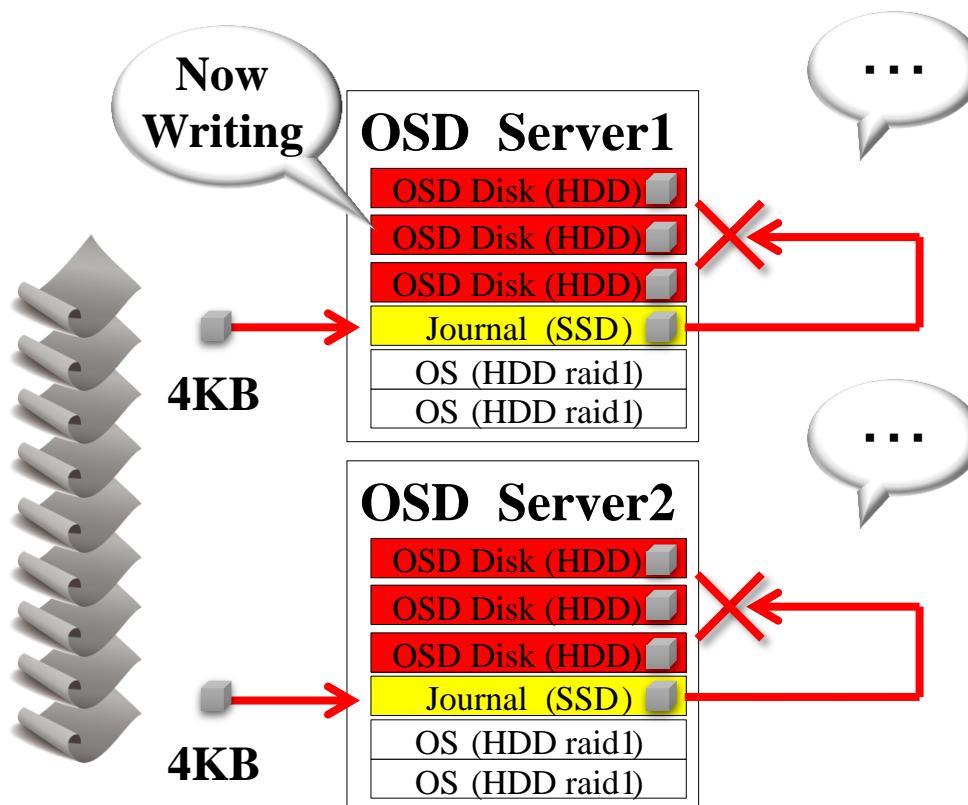
- (1) まずPrimaryがデータを受け取ったら
- (2) JournalからOSD Diskへの書き込みと同時に、
- (2') 別OSDサーバへミラーリングを指示
- (3) 別OSDサーバのJournalからOSD Diskへ書き込み

### ■ 速度が遅い原因として

- ・Diskの数の問題
  - ・キャッシュ時間の問題
- が考えられる

## 4.3 HDDが少ないからWrite速度遅いんじゃないの問題(2/4)

Disk数が少ないからWrite速度遅いんじゃないの？



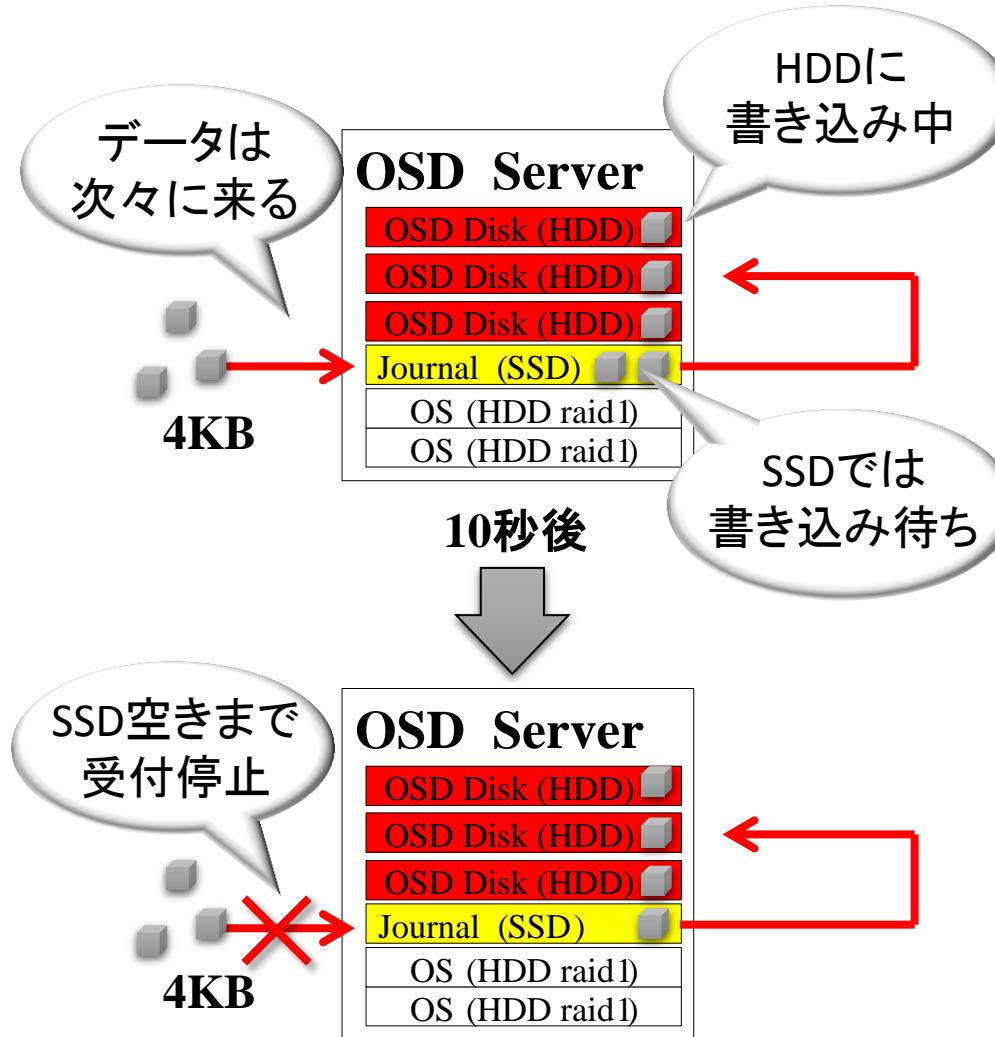
### ■ どういうこと？

Journal(SSD)へ高速に記録しても、Sync先であるOSD(HDD)が既に書き込み中だとSSDからOSD(HDD)への書き込みが進まず、**OSD(HDD)がボトルネックになる。**

高負荷環境下では結局OSD(HDD)と同速度となり、**OSD(HDD)の本数と性能が比例すると推測される。**

## 4.3 HDDが少ないからWrite速度遅いんじゃないの問題(3/4)

### キャッシング保持時間の影響でWrite速度遅いんじゃないの？



#### ■ どういうこと？

Journalにはデータのキャッシング時間を設定できる。

(中略) --filestore\_max\_sync\_interval 10

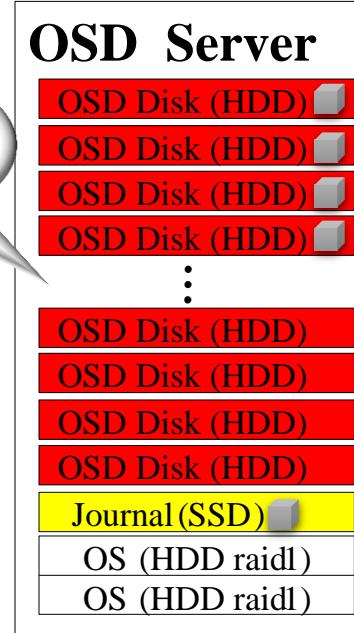
OSD(HDD)への書き込み待ちが設定時間を超えると、Journal(SSD)が空くまでデータ書き込み受付を停止。

結局OSD(HDD)の本数と性能が比例する結果に。

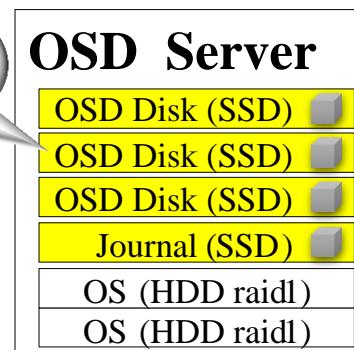
## 4.3 HDDが少ないからWrite速度遅いんじゃないの問題(4/4)

### HDD台数が少ない(~中略~) 問題の解決

HDDの  
台数増加



OSD Diskに  
SSDを使用



#### ■ OSD(HDD)の台数を大きく増やす

→ Journal(SSD)速度 < OSD(HDD)速度 × 台数

SSDの性能をHDDの台数で上回れば、  
ボトルネックは解消する。

#### ■ OSD DiskとしてSSDを導入

→ Journal(SSD)速度 < OSD(SSD)速度 × 台数

OSD Diskの性能がJournalと同等なら、  
小規模環境でもボトルネックにならない。

# Ceph性能測定TIPS

- 
- 4.1 Ceph DeployによるCeph + OpenStack連携かなり大変だよ問題  
→手入力がとても多いから可能な限りツールを使おうね
  - 4.2 Diskサイズを揃えないと性能測定の結果がバラつくよ問題  
→性能の安定を重視するなら同じDiskを選ぼうね
  - 4.3 HDD台数が少ないからWrite速度遅いんじゃないの問題  
→高負荷環境では多数のHDDを用意、またはSSD用意しそうね
  - 4.4 まとめ

## 4.4 まとめ

### ■OpenStack on Ceph検証から得られたこと

#### ・今回の検証の範囲(連続的な負荷をかけた場合)では、

- ✓ JournalのWrite Cacheはあまり効果無し (性能はOSD Diskの本数に依存)
- ✓ JournalのRead Cacheは絶大な効果 (メモリキャッシュの可能性も)

#### ・実機検証によるノウハウ

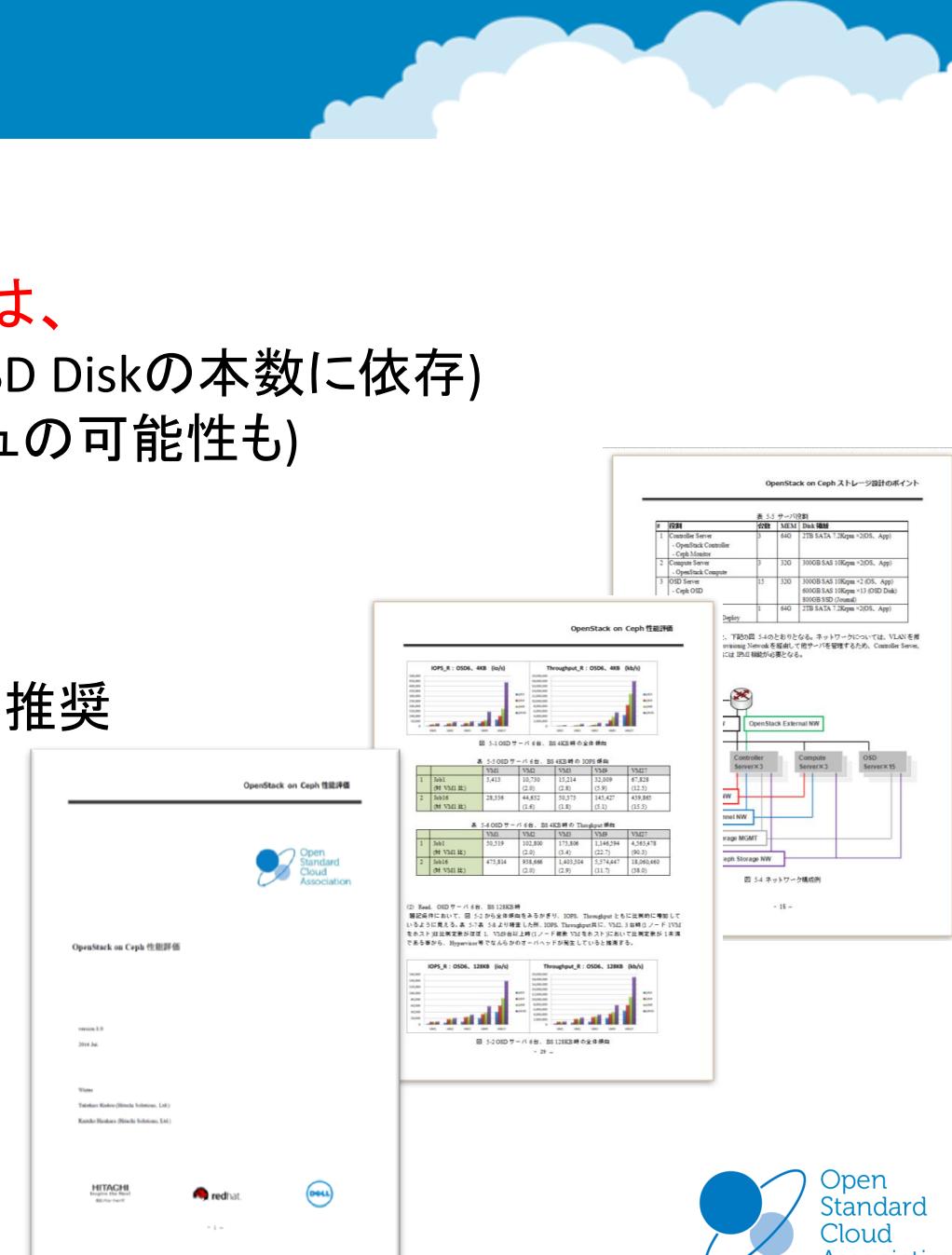
- ✓ 構築ツール推奨
- ✓ Disk統一推奨
- ✓ 小規模環境ならSSD、大規模なら多数のHDD使用を推奨

### ■OSCAホワイトペーパー発行

本日の検証内容や設計の詳細については、  
下記をご参照ください。

<http://www.osca-jp.com/solution.html>

- OpenStack on Ceph性能評価
- OpenStack on Cephストレージ設計のポイント



# 25,000のベンチマーク結果からわかった！ OpenStack on Cephの性能

OSCA技術検討会 OpenStack分科会

Linuxは、Linus Torvaldsの米国およびその他の国における登録商標または商標です。

OpenStack®の文字表記とOpenStackのロゴは、米国とその他の国におけるOpenStack Foundationの登録商標/サービスマークまたは商標/サービスマークのいずれかであり、OpenStack Foundationの許諾を得て使用しています。日立製作所はOpenStack FoundationやOpenStackコミュニティの関連企業ではなく、また支援や出資を受けていません。デルはOpenStack Foundation またはOpenStack コミュニティとは提携しておらず、公認や出資も受けていません。Red Hat は、OpenStack Foundation と OpenStack コミュニティのいずれにも所属しておらず、公認や出資も受けていません。

OSCA™ (Open Standard Cloud Association) は、デル株式会社の登録商標です。

PowerEdge、DELLロゴは、米国Dell Inc.の商標または登録商標です。

Red HatおよびRed Hatをベースとしたすべての商標とロゴ、Cephは、米国Red Hat software,Inc.の登録商標です。  
その他、記載の商標やロゴは、各社の商標または登録商標です。

本講演は、情報提供のみを目的としており、誤字脱字、技術上の誤りには一切責任を負いません。  
本講演の内容は一般的な原則を記しており、すべての環境での動作を保証するものではありません。  
本講演の内容は検証時のものであり、明示的、暗示的を問わず、いかなる内容も保証いたしません。  
本文書に掲載された文章、画像、図面等は、特に記載がない限り、OSCA、日立ソリューションズ、  
レッドハット、デルが保有しています。特に記載がない限り、複製、改変したものを無断で  
再配布することはできません。

