Storage management in a large scale at Brookhaven National Laboratory

Authors:
Robert Hancock
Technology Engineer
Brookhaven National Laboratory
Hancock@bnl.gov

Contributions: Guangwei Che, Tim Chou, Hironori Ito, Zhenping Liu, Ognian Novakov, Yingzi Wu, Tejas Rao

BROOKHAVEN NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

Outline

- Hardware Raid Enclosure
  - Description
  - Strengths and weaknesses
  - Performance
- JBOD Enclosure
  - Description
  - Strengths and Weaknesses
  - Performance
  - Initialization
  - Repair
    - Find the failed disk
    - Evaluate disk condition
    - Find the SAS address/find the physical disk
    - Replace the disk
    - Rebuild the array
  - Automate the process for time and sanity.
    - Current JBOD management software status
    - Future plans
    - ./show_disks.py
  - Questions?

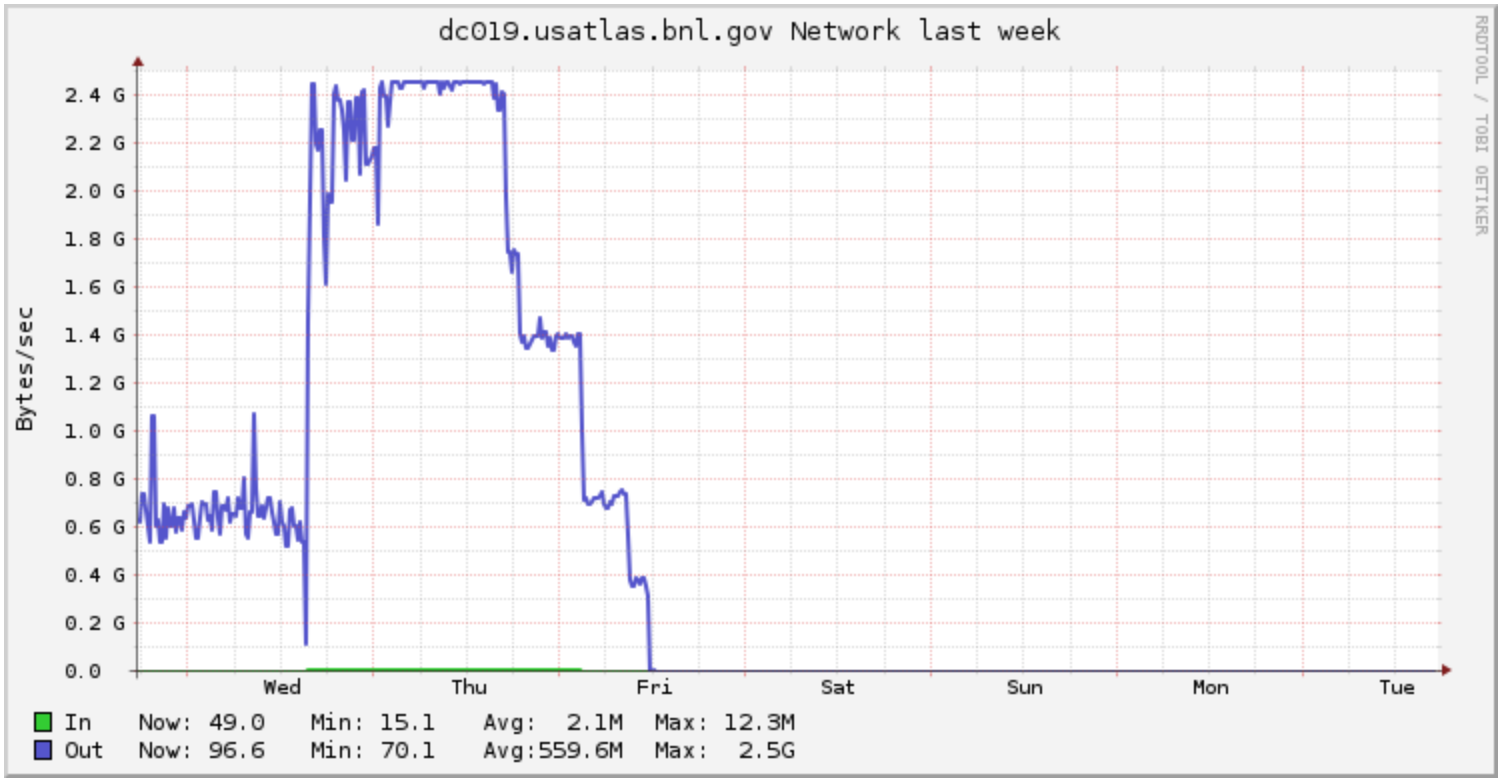
Hardware Raid Enclosure - Description

- Consists of one or more drive chassis with SAS expanders connected to a hardware raid controller.
- Connection between drive chassis and controllers commonly established with SF-8088 Mini-SAS external connectors and cables.
- Controllers commonly connected to servers that mount the raid arrays by fiber channel.
- Raid arrays appear as a single SCSI device to the host server.
- Can come with an additional management unit to oversee multiple controllers and provide a consolidated web interface for drive and chassis equipment maintenance.
- Management unit can be integrated into disk chassis or separate and connected via IP network.

Hardware Raid Enclosure – Strengths and Weaknesses

- High cost (6-11 cents/GB)
- High reliability
- Less complex administration
- Performance: good, but varied by configuration
- Low maintenance
- Examples:
  - NetApp 5760
  - Hitachi VSP G400
  - Hitachi HUS 130

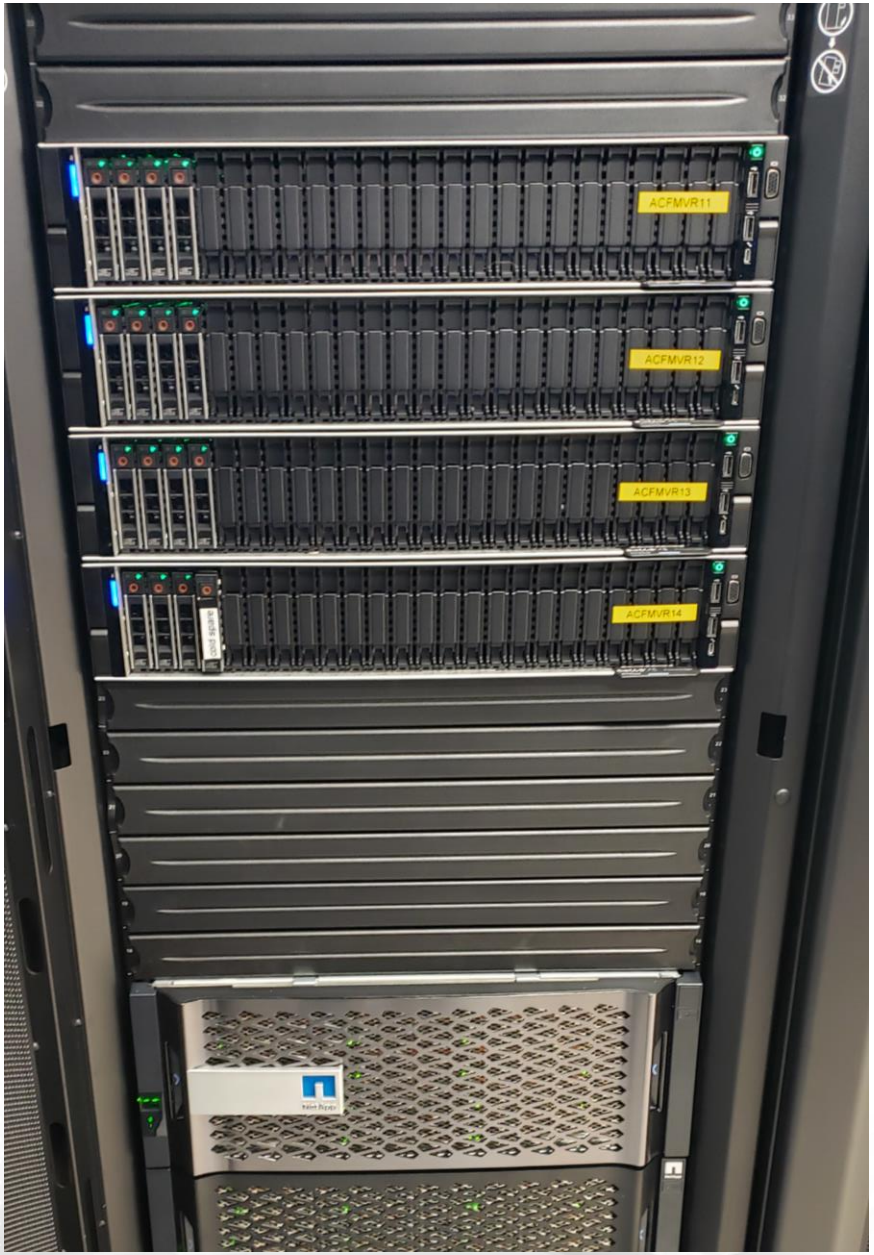
HUS 130 Performance

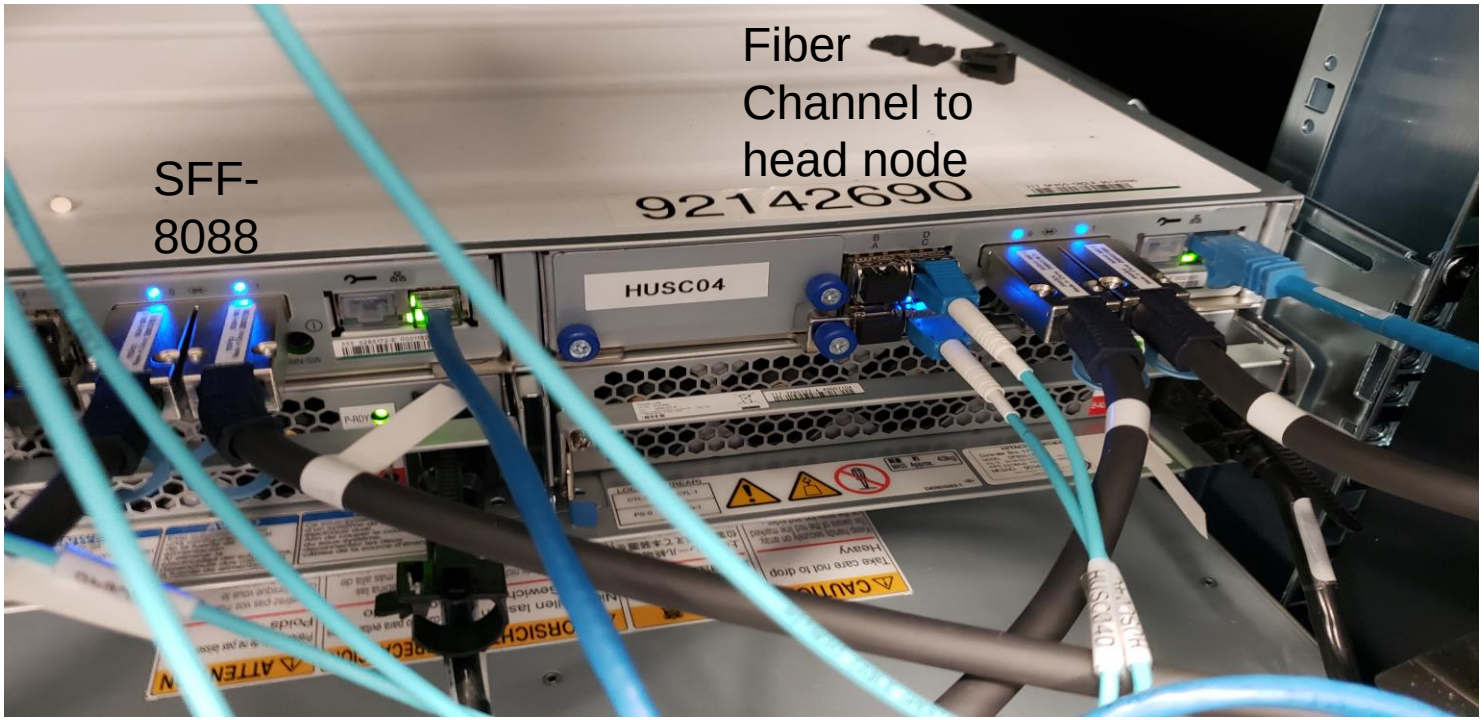


Hitachi HUS 130

Hitachi VSP G400

NetApp




Fiber Channel to head node
SFF-8088
92142690
HUSC04

Just a Bunch of Disks (JBOD) Enclosure - Description

- Consists of a disk chassis in one of many different bay counts including 84, 102, 104.
- The chassis usually includes redundant IO modules and power supplies.
- The chassis can include SAS expanders so that one SAS port on a server may communicate with many different SAS controllers connected to disk drives. This is essentially just a network switch.
- Contains enclosure services controllers which can be queried for information.

Redundant wiring for SAS expander

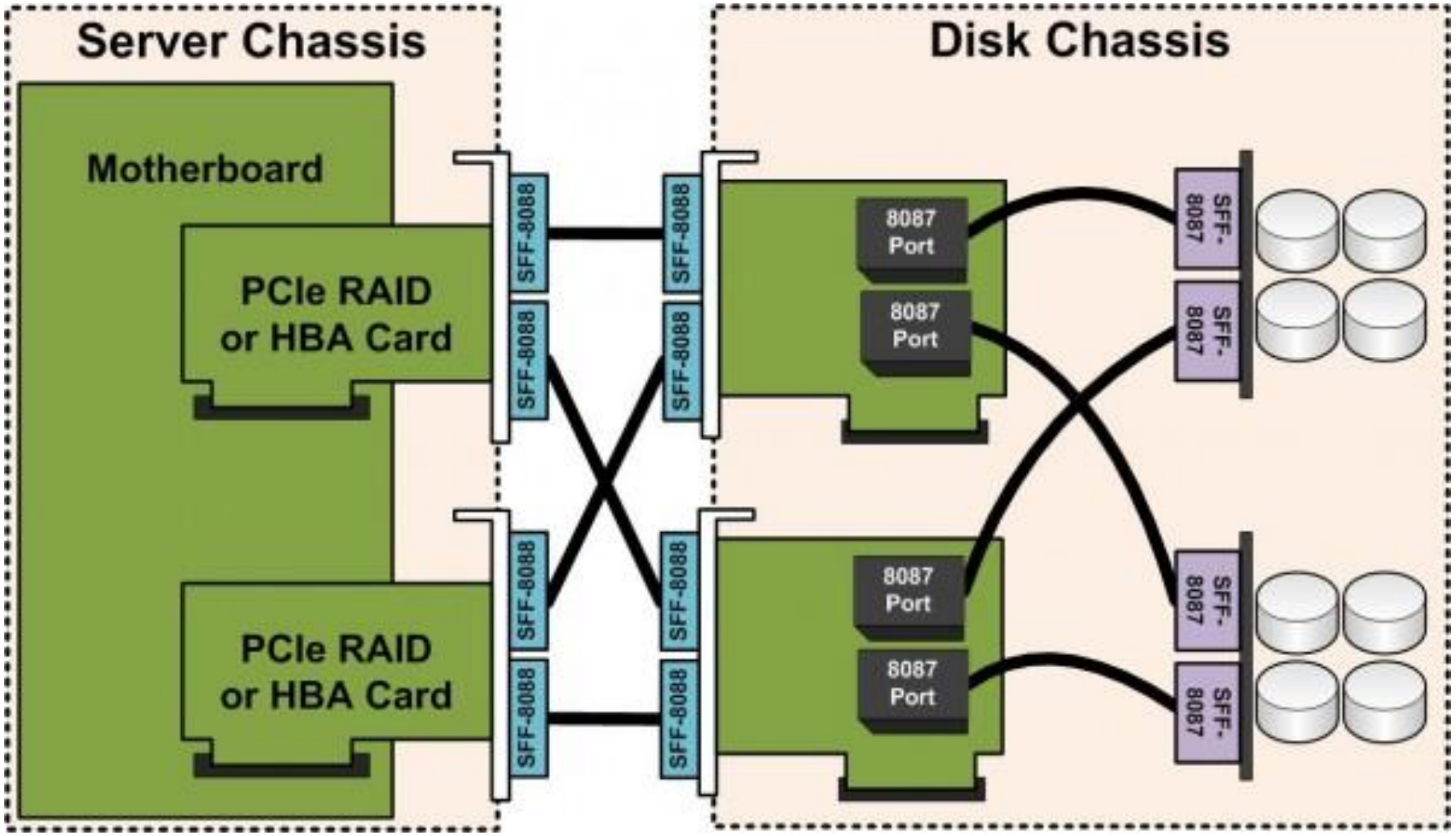
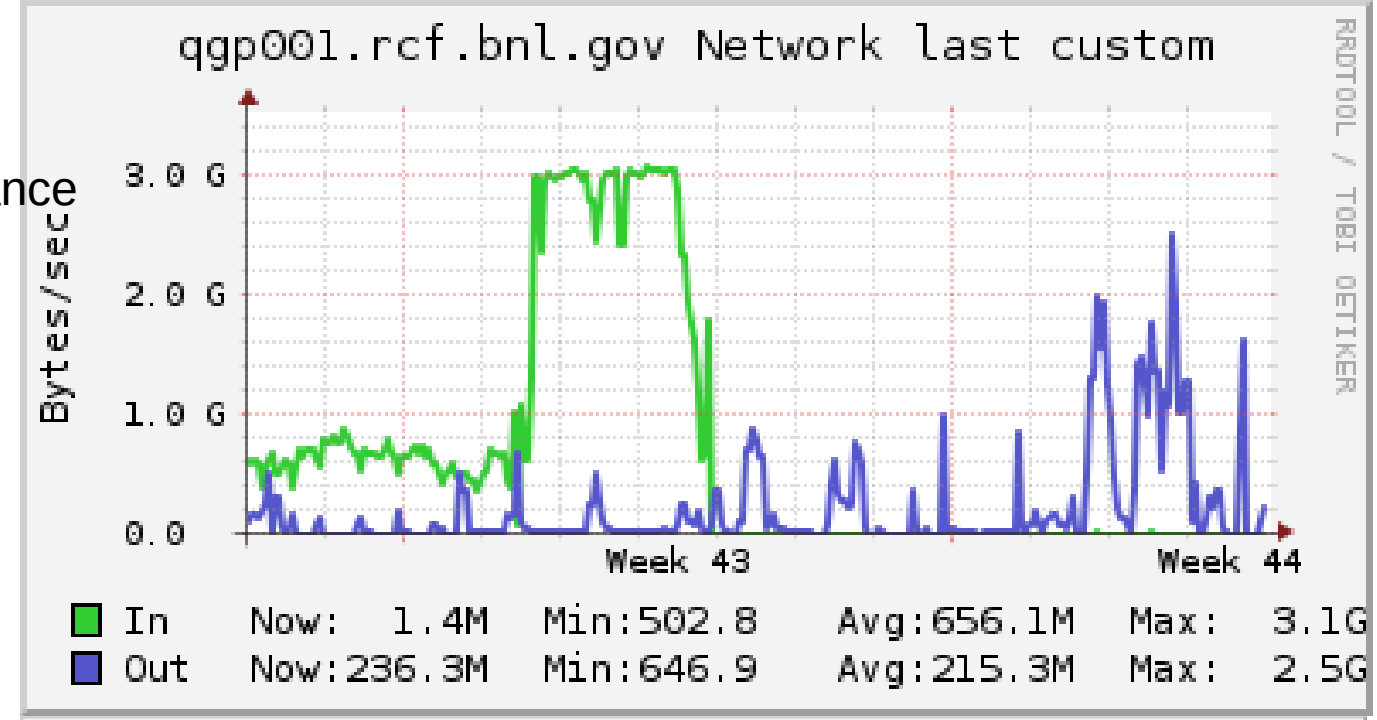


Image Credit: www.servethehome.com

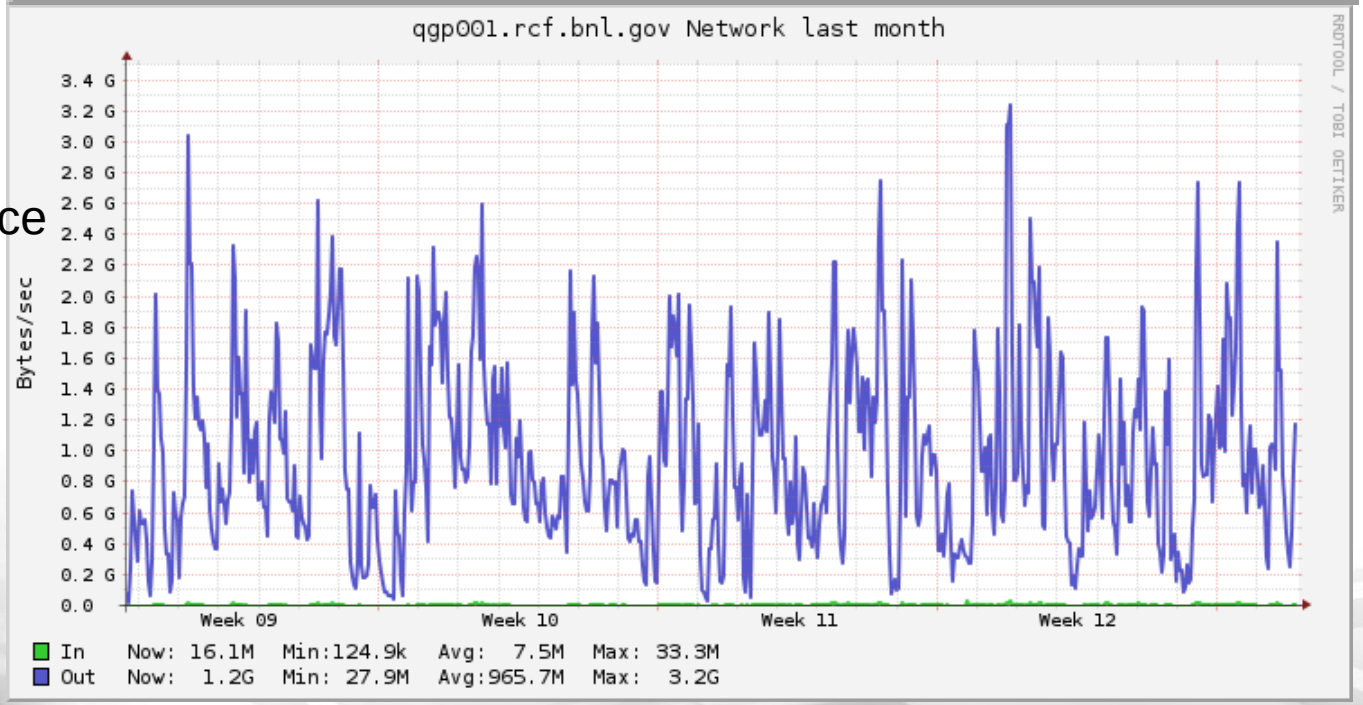
JBOD Enclosure – Strength and Weaknesses

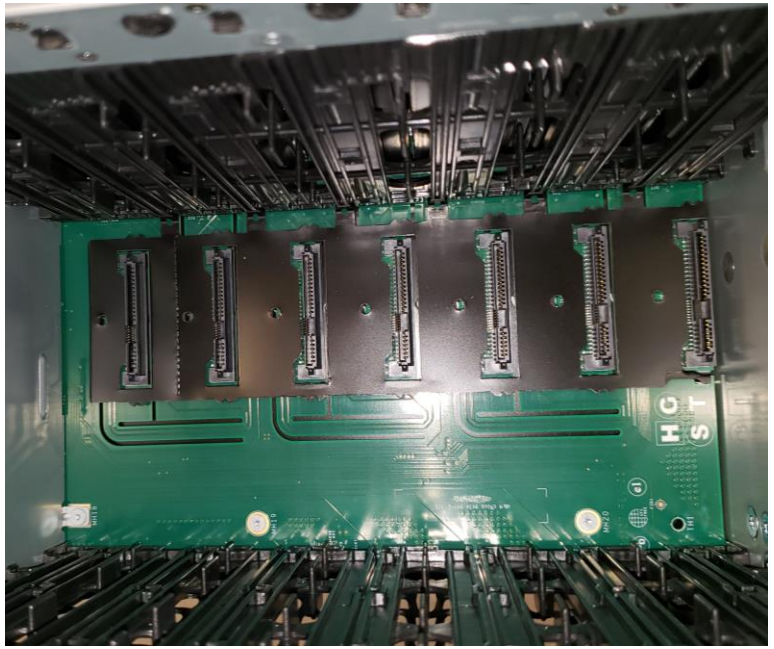
- Relatively inexpensive (~4 cents/GB)
- High Reliability
- Very complex administration
- Higher maintenance (man hours)
- Good performance (requires tuning to workload)

JBOD write performance

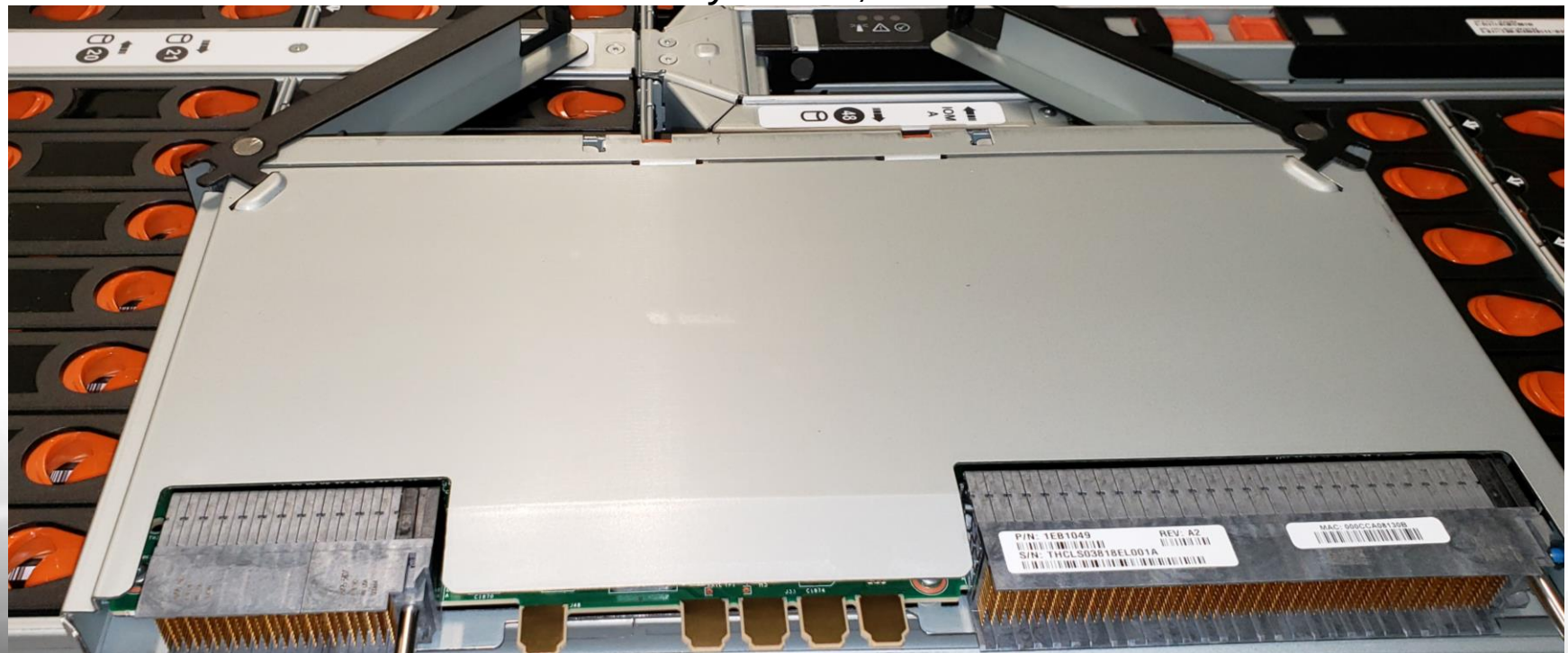


JBOD read performance

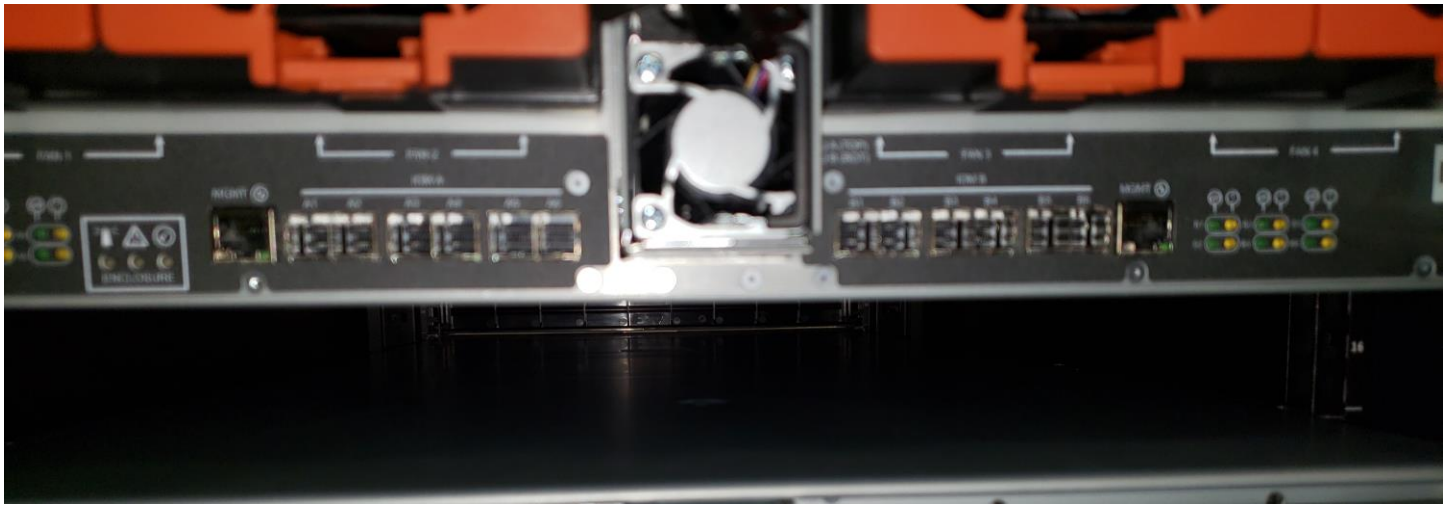




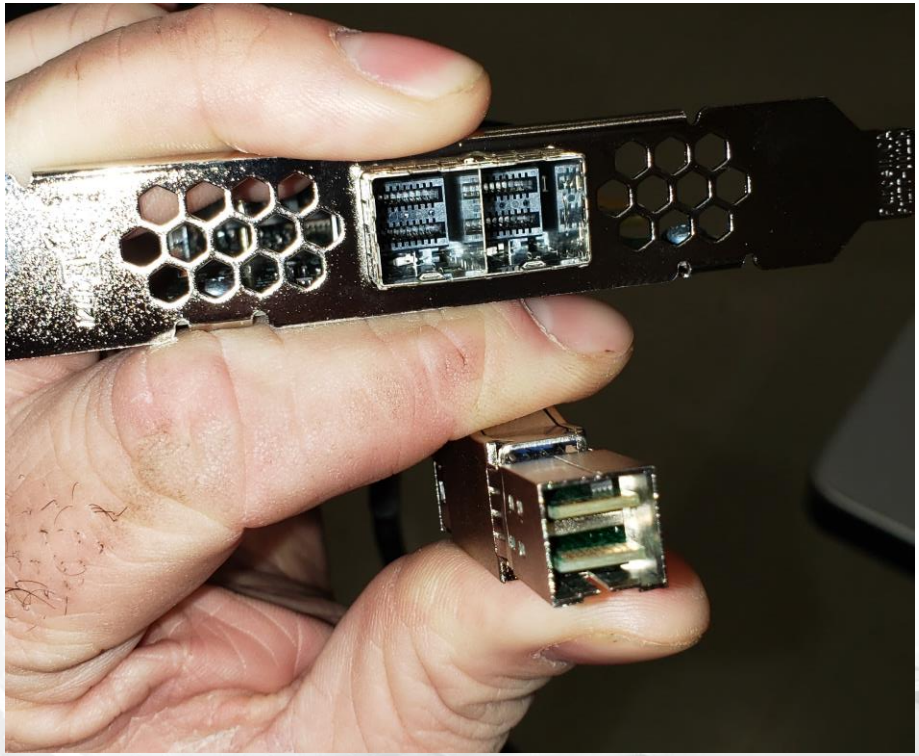
RAID INC 102 Bay JBOD, 12 TB x102 =1.2 PB raw.



Redundant IO modules, SFF-8644 12 Gbps



SFF-8644



JBOD Enclosure - Initialization

- Operating system “sees” 1 disk for each path from the server to the drive. The fully redundant wiring diagram from above would show 408 drives
- The multipathd service is used to collate these into unique “dm” devices with a 1 to 1 correspondence to physical disks.
- Each unique drive is prepared by creating a GPT label and a raid partition.
- Separate the unique drives into LUNs and use mdadm to create the arrays. (We do 12+2 raid 6 on these JBODs)
- Check the status of the new arrays with cat /proc/mdstat

Happy Raid Devices

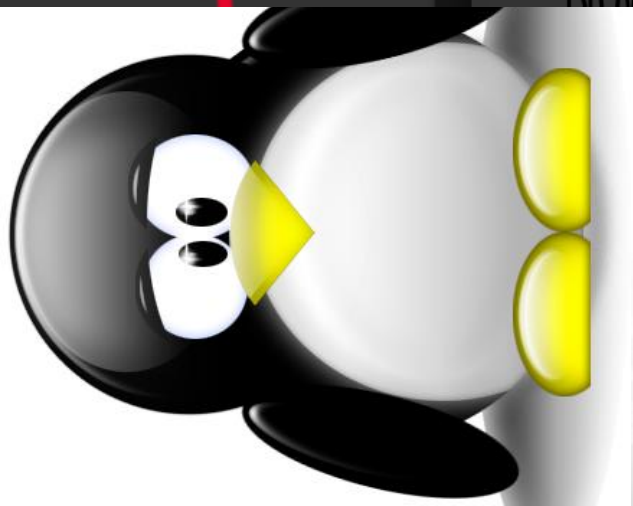
```
[root@qgp002 ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid6 dm-112[17](S) dm-107[16](S) dm-104[15](S) dm-99[14](S) dm-109[0] dm-134[13]
n-130[10] dm-136[9] dm-124[8] dm-126[7] dm-122[6] dm-119[5] dm-128[4] dm-118[3] dm-114[2] dm-1
      140625014784 blocks super 1.2 level 6, 512k chunk, algorithm 2 [14/14] [UUUUUUUUUUUUUU]
      bitmap: 2/350 pages [8KB], 16384KB chunk, file: /bitmap/md0

md6 : active raid6 dm-69[0] dm-101[13] dm-95[12] dm-97[11] dm-90[10] dm-89[9] dm-92[8] dm-86[7
3[4] dm-84[3] dm-77[2] dm-73[1]
      140625014784 blocks super 1.2 level 6, 512k chunk, algorithm 2 [14/14] [UUUUUUUUUUUUUU]
      bitmap: 0/350 pages [0KB], 16384KB chunk, file: /bitmap/md6

md4 : active raid6 dm-13[0] dm-43[13] dm-39[12] dm-41[11] dm-37[10] dm-35[9] dm-32[8] dm-31[7]
[4] dm-21[3] dm-17[2] dm-15[1]
      140625014784 blocks super 1.2 level 6, 512k chunk, algorithm 2 [14/14] [UUUUUUUUUUUUUU]
      bitmap: 0/350 pages [0KB], 16384KB chunk, file: /bitmap/md4
```

Some time later…OH NO!

```
Personalities : [raid6] [raid5] [raid4]
md5 : active raid6 dm-319[15] dm-273[14] dm-340[13] dm-307[11] dm-294[10] dm-285[9] dm-278[8] dm-248[6] dm-241[5] dm-228[4] dm-224[3] dm-208[2] dm-198[1](F)
      93766717440 blocks super 1.2 level 6, 512k chunk, algorithm 2 [14/13] [U_UUUUUUUUUUU]
      bitmap: 466/466 pages [1864KB], 8192KB chunk, file: /bitmap/md5
```

Now you have raid device simil

md5 has a problem



JBOD Enclosure - Repair

[root@dc042 ~]# cd /sys/block/dm-198/slaves/
[root@dc042 slaves]# ls
dm-191
[root@dc042 slaves]# cd dm-191/slaves/
[root@dc042 slaves]# ls
sddh sdix
----------Use smartctl, grep through /var/log/messages, query the drive with various commands from the sg_utils package…Once you decide the drive does in fact need to be replaced…---------
[root@dc042 slaves]# cd sddh/
bdi/ device/ holders/ power/ queue/ sddh1/ slaves/ subsystem/ trace/
[root@dc042 slaves]# cd sddh/device/
[root@dc042 device]# cat sas_address
0x5000c5007b300325
(alternatively you can use ‘lsscsi –gt | grep sddh’ to get sas_address)

JBOD Enclosure - Repair

- Now you need to match that sas_address with information from the enclosure services controllers to find the physical slot number.

```
[root@dc042 device]# lsscsi -gt | grep encl
[1:0:0:0]    enclosu sas:0x500093d000f4002d          -          /dev/sg1
[10:0:0:0]   enclosu sas:0x500093d00101802d          -          /dev/sg87
[12:0:0:0]   enclosu sas:0x500093d000f4022d          -          /dev/sg172
[13:0:0:0]   enclosu sas:0x500093d00101822d          -          /dev/sg257
for i in {1,87,172,257}; do echo '-----sg'$i'------'; sg_ses --join /dev/sg$i | egrep -A5 -
B20 '5000c5007b300325'; done
```

-----sg1------
-----sg87------
SLOT 80 6A [0,79] Element type: Array device slot
  Enclosure Status:
    Predicted failure=0, Disabled=0, Swap=0, status: OK
    OK=0, Reserved device=0, Hot spare=0, Cons check=0
    In crit array=0, In failed array=0, Rebuild/remap=0, R/R abort=0
    App client bypass A=0, Do not remove=0, Enc bypass A=0, Enc bypass B=0
    Ready to insert=0, RMV=0, Ident=0, Report=0
    App client bypass B=0, Fault sensed=0, Fault reqstd=0, Device off=0
    Bypassed A=0, Bypassed B=0, Dev bypassed A=0, Dev bypassed B=0
  Additional Element Status:
    Transport protocol: SAS
    number of phys: 1, not all phys: 1, device slot number: 80
    phy index: 0
      device type: end device
      initiator port for:
      target port for: SSP
      attached SAS address: 0x500093d001018180
      SAS address: 0x5000c5007b300325
      phy identifier: 0x1f

- So our failed drive lives in slot 80 of the chassis that has enclosure services controller /dev/sg87.
- We have to set the LED of the drive to blinking so we can locate and replace it.

sg_ses –index=0,79 –set 3:5:1 /dev/sg87
sg_ses –index=0,79 –set Ident /dev/sg87

- Also use mdadm, and multipath commands to remove this drive from its md device and flush its multipath map.
- Now we are back to same condition as a hardware raid when a drive fails.
- Once we replace the drive we have to to prepare the new drive to join the array
  - Create new gpt label
  - Create new raid partition
  - Use ‘mdadm –a …’ to add the new device to md.
- The array will rebuild and business continues.
- Now I actually glossed over several of the steps and difficulties that can be encountered in this process in the interests time.
- The GOOD NEWS: most of this can be, and has been, automated. I have been working on a program for a few months that uses several python classes to automate everything from initial raid creation, raid wiping, disk replacement, and health monitoring.

Automate the process for time and sanity with JBOD Management software

- The goal is to present a web interface that provides a graphical representation of each JBOD chassis. The status of each disk, path, and raid array related to the chassis are available at a glance.
- More detailed information can be accessed with only a few clicks
- Drive evaluation and replacement operations accessible from context menus.
- After a few clicks, you take an optional printout with relevant information for physically locating the disk, and go replace it.
- After the disk is replaced the program detects this offers to rebuild the array for the administrator.

Current status:

- Majority of underlying python classes exist and have been in use at BNL
  - There are classes which represent each physical and logical device and have functions designed to gather information unique to each device on instantiation.
  - Another class creates and inventory of each type of device on the system: SES enclosure controllers, hardware raid controllers, scsi disks, multipath devices, md devices …
- System inventory and status information is written into sqlite3 database
- Web GUI construction is just beginning but some useful command line programs exist. Such as show_disks.py

Enclosures inside the chassis---------- 500093d000f40000 ----------

----------Start of disks in enclosure sg172 ----------

Index	Slot	Disk Name	Multipath Device	Mpath Partition	md raid
0	1	sdhw	dm-157/mpathep	dm-170/mpathep1	md6
1	2	sdic	dm-121/mpathev	dm-131/mpathev1	md6
2	3	sdii	dm-135/mpathfb	dm-144/mpathfb1	md6
3	4	sdcy	dm-143/mpathfn	dm-154/mpathfn1	md6
4	5	sdfo	dm-6/mpathch	dm-25/mpathch1	md6
5	6	sdfu	dm-52/mpathcn	dm-63/mpathcn1	md6
6	7	sdga	dm-8/mpathct	dm-17/mpathct1	md6
7	8	sdgg	dm-20/mpathcz	dm-33/mpathcz1	md6
8	9	sdgm	dm-35/mpathdf	dm-44/mpathdf1	md6
9	10	sdgs	dm-90/mpathdl	dm-99/mpathdl1	md6
10	11	sdgy	dm-111/mpathdr	dm-129/mpathdr1	md6
11	12	sdhe	dm-87/mpathdx	dm-100/mpathdx1	md6
12	13	sday	dm-68/mpathed	dm-78/mpathed1	md6
13	14	sdhq	dm-98/mpathej	dm-110/mpathej1	md6
14	15	sdhv	dm-159/mpatheo	dm-169/mpatheo1	md7
15	16	sdib	dm-127/mpatheu	dm-133/mpatheu1	md7
16	17	sdih	dm-120/mpathfa	dm-130/mpathfa1	md7
17	18	sdin	dm-132/mpathfg	dm-150/mpathfg1	md7

Questions? Comments? `complaints -> /GARBAGE/`