# 测试对象

1. **测试对象**

HaiShen OCSSD上建立的类型为pblk的target。

1. **参数设置**
2. PU数量：

通过liblightNVM的CLI可知haishen OCSSD总共有32个PU，在使用nvme-cli建立pblk的target时，选取全部32个PU。

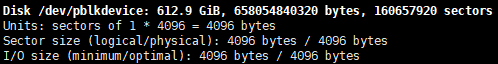
1. Target名称：

pblkdevice。

1. **对象建立命令**

nvme lnvm create –d nvme0n1 –n pblkdevice –b 0 –e 31 –t pblk -f

1. **对象容量**



# 测试目的

了解HaiShen OCSSD以pblk类型来管理的性能表现。

# 测试工具

使用FIO来进行各种负载下的性能测试。

# 测试实例与结果

**测试实例说明：**

本测试文档中采用的各种测试实例依据的是[Intel的QoS测试实例](https://github.com/intel/fiovisualizer/tree/master/Workloads/Device)，其测试实例根据设备类型的不同分为NVMe和SATA两类，这里选取针对NVMe设备的10种测试实例以及一些扩展实例。

每次测试实例时，target都为新建立的状态，对于读取操作，首先进行预处理（dd if=/dev/zero bs=1024k of=/dev/pblkdevice），即向盘内写入数据后再测试，短期测试时间为5min，长期测试时间为1h。

1. **4k的随机混合读写（队列深度为4，进程数为32）**
2. 测试参数设置及说明（ps：后续实例的参数设置及说明不再赘述）

[global]

name=4k random MIX 70/30 with 4 ios in the queue in 32 queues #实例名称

filename=/dev/pblkdevice #测试设备的名称

ioengine=libaio #使用异步IO模式

direct=1 #使用直连directIO

bs=4k #产生IO单元大小为4k

rw=randrw #使用IO类型为随机混合读写

rwmixread=70 #混合读写中，读占70%

iodepth=4 #队列深度为4

numjobs=32 #为job副本指定32个进程

buffered=0 #不使用buffered IO

size=100% #总共传输的数据大小为设备的总的空间

runtime=300 #控制fio在5分钟之后退出执行

time\_based #设置为要执行完runtime规定的时间

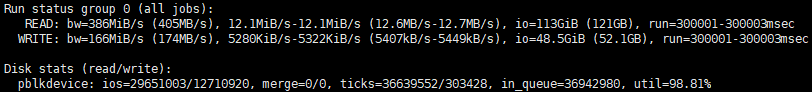
randrepeat=0 #表示这次的随机操作是不可重复的

norandommap #表示使用的是真随机的方式

refill\_buffers #表示每次submit之后会重新填满IO buffer

[job1] #

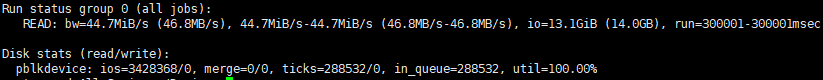
1. 测试运行截图（满盘情况下测试）



1. 测试结论

从上图可知，在4k的随机混合读写（读占70%）负载中，32个进程在队列深度为4的情况下，总的读带宽为405MB/s，总的写带宽为174MB/s，可计算出总的读IOPS为405 \* 1024 / 4 = 103680，总的写IOPS为174 \* 1024 / 4 = 44544。

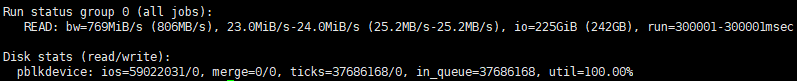
1. **4k的随机读取（队列深度为1，进程数为1）**
2. 测试运行截图



1. 测试结论

从上图可知，在4k的随机读取负载中，单个进程在队列深度为1的情况下，总的读带宽为46.8MB/s，可计算出总的读IOPS为46.8 \* 1024 / 4 = 11981。

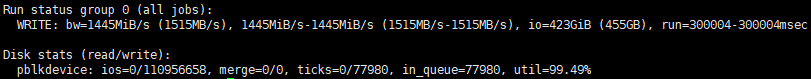
1. **4k的随机读取（队列深度为4，进程数为32）**
2. 测试运行截图



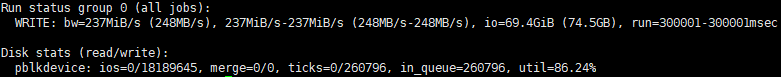
1. 测试结论

从上图可知，在4k的随机读取负载中，32个进程在队列深度为4的情况下，总的读带宽为806MB/s，可计算出总的读IOPS为806 \* 1024 / 4 = 206336。

1. **4k的随机写入（队列深度为1，进程数为1）**
2. Fob测试运行截图（空盘情况下测试）



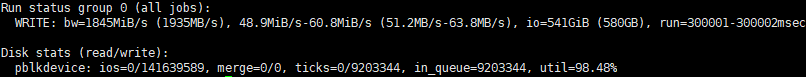
1. 稳态测试运行截图（满盘情况下测试）



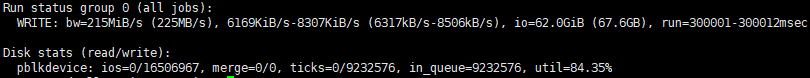
1. 测试结论

从上图可知，在4k的随机写入负载中，单个进程在队列深度为1的情况下，Fob总写带宽为1515MB/s，计算出总的写IOPS为1515 \* 1024 / 4 = 387840；稳态总写入带宽为248MB/s，计算出写IOPS为248 \* 1024 / 4 = 63488。

1. **4k的随机写入（队列深度为4，进程数为32）**
2. Fob测试运行截图（空盘情况下测试）



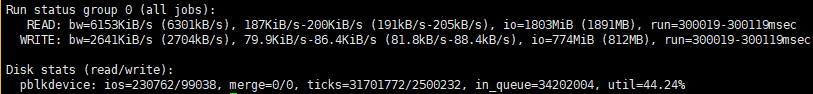
1. 稳态测试运行截图（满盘情况下测试）



1. 测试结论

从上图可知，在4k的随机写入负载中，32个进程在队列深度为4的情况下，Fob总写带宽为1935MB/s，计算出总的写IOPS为1935 \* 1024 / 4 = 495360；稳态总写入带宽为225MB/s，计算出写IOPS为225 \* 1024 / 4 = 57600。

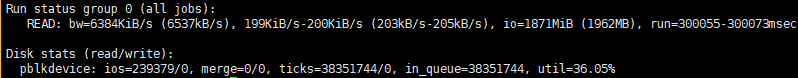
1. **8k的随机混合读写（队列深度为4，进程数为32）**
2. 测试运行截图（满盘情况下测试）



1. 测试结论

从上图可知，在8k的随机混合读写（读占70%）负载中，32个进程在队列深度为4的情况下，总的读带宽为6301kB/s，总的写带宽为2704kB/s，可计算出总的读IOPS为6301 / 8 = 788，总的写IOPS为2704 / 8 = 338。

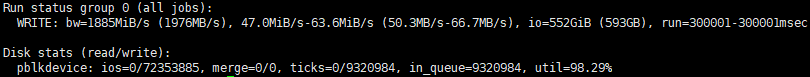
1. **8k的随机读取（队列深度为4，进程数为32）**
2. 测试运行截图



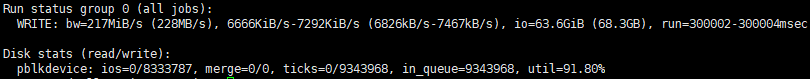
1. 测试结论

从上图可知，在8k的随机读取负载中，32个进程在队列深度为4的情况下，总的读带宽为6537kB/s，可计算出总的读IOPS为6537 / 8 = 817。

1. **8k的随机写入（队列深度为4，进程数为32）**
2. Fob测试运行截图（空盘情况下测试）



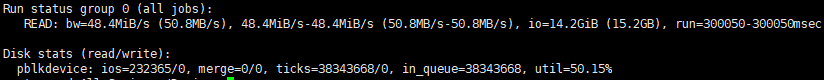
1. 稳态测试运行截图（满盘情况下测试）



1. 测试结论

从上图可知，在8k的随机写入负载中，32个进程在队列深度为4的情况下，Fob总写带宽为1976MB/s，计算出总的写IOPS为1976 \* 1024 / 8 = 252928；稳态总写入带宽为228MB/s，计算出写IOPS为228 \* 1024 / 8 = 29184。

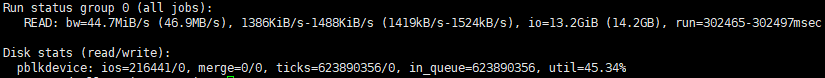
1. **64k的顺序读取（队列深度为128，进程数为1）**
2. 测试运行截图



1. 测试结论

从上图可知，在64k的顺序读取负载中，单个进程在队列深度为128的情况下，总的读带宽为50.8MB/s，可计算出总的读IOPS为50.8 \* 1024 / 64 = 813。

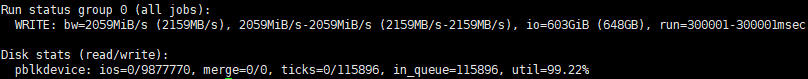
1. **64k的顺序读取（队列深度为128，进程数为32）**
2. 测试运行截图



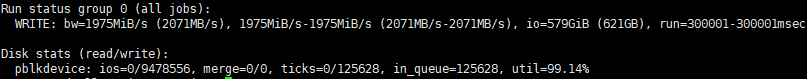
1. 测试结论

从上图可知，在64k的顺序读取负载中，单个进程在队列深度为128的情况下，总的读带宽为46.9MB/s，可计算出总的读IOPS为46.9 \* 1024 / 64 = 750。

1. **64k的顺序写入（队列深度为128，进程数为1）**
2. Fob测试运行截图（空盘情况下测试）



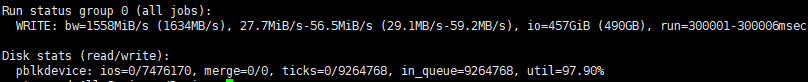
1. 稳态测试运行截图（满盘情况下测试）



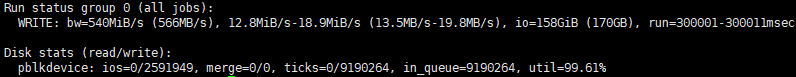
1. 测试结论

从上图可知，在64k的顺序写入负载中，单个进程在队列深度为128的情况下，Fob总写带宽为2159MB/s，计算出总的写IOPS为2159 \* 1024 / 64 = 34544；稳态总写入带宽为2071MB/s，计算出写IOPS为2071 \* 1024 / 64 = 33136。

1. **64k的顺序写入（队列深度为128，进程数为32）**
2. Fob测试运行截图（空盘情况下测试）



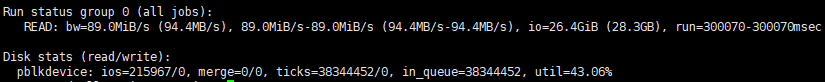
1. 稳态测试运行截图（满盘情况下测试）



1. 测试结论

从上图可知，在64k的顺序写入负载中，32个进程在队列深度为128的情况下，Fob总写带宽为1634MB/s，计算出总的写IOPS为1634 \* 1024 / 64 = 26144；稳态总写入带宽为566MB/s，计算出写IOPS为566 \* 1024 / 64 = 9056。

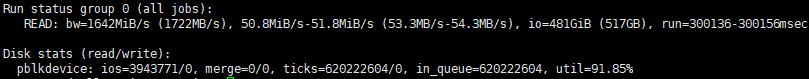
1. **128k的顺序读取（队列深度为128，进程数为1）**
2. 测试运行截图



1. 测试结论

从上图可知，在128k的顺序读取负载中，单个进程在队列深度为128的情况下，总的读带宽为94.4MB/s，这样可计算出总的读IOPS为94.4 \* 1024 / 128 = 755。

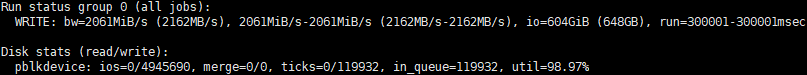
1. **128k的顺序读取（队列深度为128，进程数为32）**
2. 测试运行截图



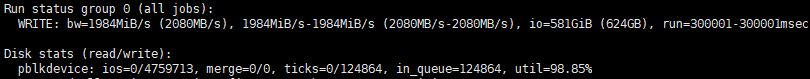
1. 测试结论

从上图可知，在128k的顺序读取负载中，32个进程在队列深度为128的情况下，总的读带宽为1722MB/s，这样可计算出总的读IOPS为1722 \* 1024 / 128 = 13776。

1. **128k的顺序写入（队列深度为128，进程数为1）**
2. Fob测试运行截图（空盘情况下测试）



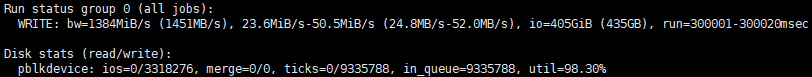
1. 稳态测试运行截图（满盘情况下测试）



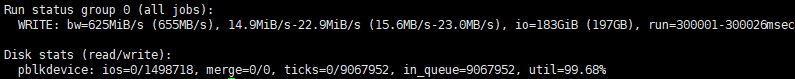
1. 测试结论

从上图可知，在128k的顺序写入负载中，单个进程在队列深度为128的情况下，Fob总写带宽为2162MB/s，计算出总的写IOPS为2162 \* 1024 / 128 = 17296；稳态总写入带宽为2080MB/s，计算出写IOPS为2080 \* 1024 / 128 = 16640。

1. **128k的顺序写入（队列深度为128，进程数为32）**
2. Fob测试运行截图（空盘情况下测试）



1. 稳态测试运行截图（满盘情况下测试）



1. 测试结论

从上图可知，在128k的顺序写入负载中，32个进程在队列深度为128的情况下，Fob总写带宽为1451MB/s，计算出总的写IOPS为1451 \* 1024 / 128 = 11608；稳态总写入带宽为655MB/s，计算出写IOPS为655 \* 1024 / 128 = 5240。

# 短期测试结果汇总（测试时间为5min）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **读写类型** | **操作类型** | **数据大小** | **进程数** | **队列深度** | **测试环境**  **(For Write)** | **带宽**  **(MB/s)** | **IOPS** |
| **MIX** | **random** | 4k | 32 | 4 |  | R:405  W:174 | R:103680  W:44544 |
| 8k | 32 | 4 |  | R:6.3  W:2.7 | R:788  W:338 |
| **Read** | **random** | 4k | 1 | 1 |  | 46.8 | 11981 |
| 32 | 4 |  | 806 | 206336 |
| 8k | 32 | 4 |  | 6.5 | 817 |
| **sequential** | 64k | 1 | 128 |  | 50.8 | 813 |
| 32 | 128 |  | 46.9 | 750 |
| 128k | 1 | 128 |  | 94.4 | 755 |
| 32 | 128 |  | 1722 | 13776 |
| **Write** | **random** | 4k | 1 | 1 | fob | 1515 | 387840 |
| stable | 248 | 63488 |
| 32 | 4 | fob | 1935 | 495360 |
| stable | 225 | 57600 |
| 8k | 32 | 4 | fob | 1976 | 252928 |
| stable | 228 | 29184 |
| **sequential** | 64k | 1 | 128 | fob | 2159 | 34544 |
| stable | 2071 | 33136 |
| 32 | 128 | fob | 1634 | 26144 |
| stable | 566 | 9056 |
| 128k | 1 | 128 | fob | 2162 | 17296 |
| stable | 2080 | 16640 |
| 32 | 128 | fob | 1451 | 11608 |
| stable | 655 | 5240 |

注：

1. MIX表示混合读写，其中读占70%。
2. fob表示使用新盘进行测试，stable表示使用写满数据的盘进行测试。
3. 红色数据表示认为不合理的地方。

# 长期测试结果汇总（测试时间为1h）

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **读写类型** | **操作类型** | **数据大小** | **进程数** | **队列深度** | **测试环境**  **(For Write)** | **带宽**  **(MB/s)** | **IOPS** |
| **MIX** | **random** | 4k | 32 | 4 |  | R:  W: | R:  W: |
| 8k | 32 | 4 |  | R:  W: | R:  W: |
| **Read** | **random** | 4k | 1 | 1 |  | 46.9 | 12006 |
| 32 | 4 |  | 806 | 206336 |
| 8k | 32 | 4 |  | 5.9 | 749 |
| **sequential** | 64k | 1 | 128 |  | 51.3 | 821 |
| 32 | 128 |  | 50.4 | 806 |
| 128k | 1 | 128 |  | 94.5 | 756 |
| 32 | 128 |  | 1874 | 14992 |
| **Write** | **random** | 4k | 1 | 1 | stable | 90 | 23040 |
| 32 | 4 | stable | 60.5 | 15488 |
| 8k | 32 | 4 | stable | 63.6 | 8141 |
| **sequential** | 64k | 1 | 128 | stable | 214 | 3424 |
| 32 | 128 | stable | 117 | 1872 |
| 128k | 1 | 128 | stable | 225 | 1800 |
| 32 | 128 | stable | 277 | 2216 |

注：

1. MIX表示混合读写，其中读占70%。
2. stable表示使用写满数据的盘进行测试。

**2019年12月重新测试数据如下所示：**

# 短期测试结果汇总（测试时间为5min）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **读写类型** | **操作类型** | **数据大小** | **进程数** | **队列深度** | **测试环境**  **(For Write)** | **带宽**  **(MB/s)** |
| **MIX** | **random** | 4k | 32 | 4 |  | R:411  W:176 |
| 8k | 32 | 4 |  | R:5.8  W:2.5 |
| **Read** | **random** | 4k | 1 | 1 |  | 41.6 |
| 1 | 4 |  | 165 |
| 32 | 4 |  | 805 |
| 8k | 1 | 1 |  | 66.6 |
| 1 | 4 |  | 262 |
| 32 | 4 |  | 6.17 |
| **sequential** | 64k | 1 | 128 |  | 42.9 |
| 32 | 128 |  | 36.2 |
| 128k | 1 | 128 |  | 99.7 |
| 32 | 128 |  | 1730 |
| **Write** | **random** | 4k | 1 | 1 | fob | 1315 |
| 1 | 4 | fob | 1309 |
| 32 | 4 | fob | 1307 |
| stable | 248 |
| 8k | 1 | 1 | fob | 1899 |
| 1 | 4 | fob | 1888 |
| 32 | 4 | fob | 1878 |
| stable | 250 |
| **sequential** | 64k | 1 | 128 | fob | 2153 |
| stable | 2139 |
| 32 | 128 | fob | 2153 |
| stable | 807 |
| 128k | 1 | 128 | fob | 2149 |
| stable | 2141 |
| 32 | 128 | fob | 2152 |
| stable | 815 |

注：

1. MIX表示混合读写，其中读占70%。
2. fob表示使用新盘进行测试，stable表示使用写满数据的盘进行测试。

# 长期测试结果汇总（测试时间为1h）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **读写类型** | **操作类型** | **数据大小** | **进程数** | **队列深度** | **测试环境**  **(For Write)** | **带宽**  **(MB/s)** |
| **MIX** | **random** | 4k | 32 | 4 |  | R:163  W:69.8 |
| 8k | 32 | 4 |  | R:5.6  W:2.39 |
| **Read** | **random** | 4k | 1 | 1 |  | 41 |
| 1 | 4 |  | 164.3 |
| 32 | 4 |  | 753 |
| 8k | 1 | 1 |  | 65.2 |
| 1 | 4 |  | 251 |
| 32 | 4 |  | 5.55 |
| **sequential** | 64k | 1 | 128 |  | 49.9 |
| 32 | 128 |  | 49.4 |
| 128k | 1 | 128 |  | 94.3 |
| 32 | 128 |  | 1978 |
| **Write** | **random** | 4k | 1 | 1 | stable | 89 |
| 1 | 4 | stable | 89.8 |
| 32 | 4 | stable | 89 |
| 8k | 1 | 1 | stable | 92.8 |
| 1 | 4 | stable | 92.7 |
| 32 | 4 | stable | 81.4 |
| **sequential** | 64k | 1 | 128 | stable | 2141 |
| 32 | 128 | stable | 589 |
| 128k | 1 | 128 | stable | 2140 |
| 32 | 128 | stable | 591 |

注：

1. MIX表示混合读写，其中读占70%。
2. stable表示使用写满数据的盘进行测试。

**使用spdk自带bdevperf工具进行测试，结果如下：**

# 短期测试结果汇总（测试时间为5min）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **读写类型** | **操作类型** | **数据大小** | **进程数** | **队列深度** | **测试环境**  **(For Write)** | **带宽**  **(MB/s)** |
| **MIX** | **random** | 4k | 1 | 4 |  | R:1228  W: |
| 8k | 1 | 4 |  | R:1502  W: |
| **Read** | **random** | 4k | 1 | 1 |  | 34.28 |
| 1 | 4 |  | 82.87 |
| 8k | 1 | 1 |  | 61.31 |
| 1 | 4 |  | 158.13 |
| **sequential** | 64k | 1 | 128 |  | 1220.17 |
| 128k | 1 | 128 |  | 1309.88 |
| **Write** | **random** | 4k | 1 | 1 | fob | 1375.16 |
| 1 | 4 | fob | 1341.23 |
| 8k | 1 | 1 | fob | 1377.39 |
| 1 | 4 | fob | 1354.18 |
| **sequential** | 64k | 1 | 128 | fob | 1376.46 |
| 128k | 1 | 128 | fob | 1380.37 |

注：

1. MIX表示混合读写，其中读占70%。
2. fob表示使用新盘进行测试，stable表示使用写满数据的盘进行测试。

# 长期测试结果汇总（测试时间为1h）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **读写类型** | **操作类型** | **数据大小** | **进程数** | **队列深度** | **测试环境**  **(For Write)** | **带宽**  **(MB/s)** |
| **MIX** | **random** | 4k | 1 | 4 |  | R:738  W: |
| 8k | 1 | 4 |  | R:824  W: |
| **Read** | **random** | 4k | 1 | 1 |  | 34.13 |
| 1 | 4 |  | 79.53 |
| 8k | 1 | 1 |  | 61.28 |
| 1 | 4 |  | 153.51 |
| **sequential** | 64k | 1 | 128 |  | 1253.64 |
| 128k | 1 | 128 |  | 1256.68 |
| **Write** | **random** | 4k | 1 | 1 | stable | 570.15 |
| 1 | 4 | stable | 565.16 |
| 8k | 1 | 1 | stable | 787.57 |
| 1 | 4 | stable | 782.47 |
| **sequential** | 64k | 1 | 128 | stable | 1351.92 |
| 128k | 1 | 128 | stable | 1348.15 |

注：

1. MIX表示混合读写，其中读占70%。
2. stable表示使用写满数据的盘进行测试。