InfoSphere VDP Tech Brief

Using IOPERF to Ensure Allocated Storage Meets VDP Requirements

Purpose

The purpose of this document is to provide the IBM InfoSphere representative a tool to determine whether the storage allocated to an IBM InfoSphere storage pool can support the demands placed on it.

All scripts and binaries for this procedure reside under `/act/sky/ioperf/bin` and `/act/sky/ioperf/scripts`

CAUTION! *This procedure overwrites the storage that it evaluates. You MUST run this procedure on storage that is either not in use or contains data that is expendable.*

Procedure

After the VDP Appliance ova, vhdx, or AMI has been deployed and powered on, but prior to actually running the IBM InfoSphere installer:

1. Log into the VDP Appliance virtual machine via a terminal session; using port 26, root user, and AVIN key.

2. Create a 200GB disk on the same storage as the Dedup Pool.

    Run `lsblk` and `grep` the 200GB size of the storage and get the DiskName of the newly created hard disk that you created on the Dedup Pool storage. For example:

    [root@NVPS actifio]# lsblk | grep 200G

    ```
    mydiskname 202:80   0 200G  0 disk
    ```

    In this example, the disk name is **mydiskname**.

3. Use `fdisk` to find the number of available bytes on the disk:

    [root@localhost ~]# fdisk -l /dev/<diskname> | grep Disk| grep -v identifier| cut -d " " -f 5

    ```
    214748364800
    ```

4. Set the environment variable:

    export IOPERF_BINARY=/act/sky/ioperf/bin/ioperf

5. Make a directory in which to run scripts and to put the script's output. For example:

    mkdir /var/log/perftest

6.   Change to that directory:

cd /var/log/perftest

7. Run nohup with the disk name and size. For example:

nohup /act/sky/ioperf/scripts/ioperf_fill_run_sky.py <diskname> 214748364800 &

nohup with & will run the command in background mode which can be viewed and managed by jobs and `fg`

The script will run four tests and create log files for each test.

`rand-reader-2016-11-16-08:29:24.csv`

`rand-writer-2016-11-16-08:29:24.csv`

`seq-reader-2016-11-16-08:29:24.csv`

`seq-writer-2016-11-16-08:29:24.csv`

Note: *The log files will be created in the directory from which you executed the script. In this example it will be:* `/var/log/perftest`

8. To determine if the storage is compatible with the VDP Appliance, open and examine each log file and review the threads tested to determine if they meet the following requirements:

`output-rand-reader.csv`

Each thread requires a minimum of 60 Ops per second. For example a single thread requires a minimum of 60 Ops per second. To maintain two threads, 120 Ops per second is required. Three threads require at least 180 Ops per second. The following example shows 15 threads, each of which do not meet the minimum requirements:

```
Threads, Time(s), MB/s, Ops/s, Min_Ops/s
    15 , 467 ,   16.684 , 266, 900
    14 , 435 ,   15.349 , 245, 840
    13 , 402 ,   15.772 , 252, 780
    12 , 372 ,   15.443 , 247, 720
    11 , 342 ,   13.748 , 219, 660
    10 , 307 ,   11.836 , 189, 600
     9 , 282 ,   10.819 , 173, 540
     8 , 248 ,   11.275 , 180, 480
     7 , 218 ,    9.684 , 154, 420
     6 , 185 ,    8.967 , 143, 360
     5 , 153 ,    7.829 , 125, 300
     4 , 122 ,    6.174 ,  98, 240
     3 ,  92 ,    4.570 ,  73, 180
     2 ,  61 ,    3.254 ,  52, 120
     1 ,  30 ,    1.683 ,  26,  60
```

`output-rand-writer.csv`

Each of the first four threads must have a minimum of 200 Ops per second. In the following example, the first three threads do not meet this requirement.

```
Threads, Time(s), MB/s, Ops/s, Min_Ops/s
    10 ,  309 , 17.349 , 277,   200
     9 ,  276 , 16.483 , 263,   200
     8 ,  246 , 17.966 , 287,   200
     7 ,  215 , 15.604 , 249,   200
     6 ,  185 , 16.069 , 257,   200
     5 ,  154 , 15.455 , 247,   200
     4 ,  123 , 12.757 , 204,   200
     3 ,   93 ,  9.738 , 155,   200
     2 ,   62 ,  7.154 , 114,   200
     1 ,   31 ,  3.338 ,  53,   200
```

`output-seq-reader.csv`

Each of the first four threads must have a minimum of 2000 Ops per second. In the following example, the first three threads do not meet this requirement.

```
Threads, Time(s), MB/s, Ops/s, Min_Ops/s
  10 ,   311 , 150.813 , 2413, 2000
   9 ,   279 , 146.750 , 2348, 2000
   8 ,   248 , 143.567 , 2297, 2000
   7 ,   216 , 122.843 , 1965, 2000
   6 ,   185 , 127.498 , 2039, 2000
   5 ,   153 , 133.120 , 2129, 2000
   4 ,   123 , 130.116 , 2081, 2000
   3 ,    91 , 104.246 , 1667, 2000
   2 ,    62 ,  96.766 , 1548, 2000
   1 ,    31 ,  56.502 ,  904, 2000
```

output-seq-writer.csv 2000 ops for threads 1 to 4

Each of the first four threads must have a minimum of 2000 Ops per second. In the following example, the first three threads do not meet this requirement.

```
Threads, Time(s), MB/s, Ops/s, Min_Ops/s
  10 ,   310 , 52.003 , 832,   2000
   9 ,   273 , 48.363 , 773,   2000
   8 ,   235 , 44.321 , 709,   2000
   7 ,   212 , 47.041 , 752,   2000
   6 ,   186 , 46.093 , 737,   2000
   5 ,   151 , 48.773 , 780,   2000
   4 ,   120 , 57.875 , 926,   2000
   3 ,    92 , 50.512 , 808,   2000
   2 ,    61 , 41.846 , 669,   2000
   1 ,    30 , 25.755 , 412,   2000
```

9. Zip and save each csv file for future reference.