PURESTORAGE

Best Practices for Oracle on Pure Storage

The First All-Flash Enterprise Storage Array

purestorage.com

Overview

The principle difference between configuring database storage on a Pure Storage FlashArray™ instead of spinning disks is that virtually all of your architecture choices are centered around manageability, not performance. Specifically, none of the following factors are relevant on a Pure Storage array:

- Stripe width and depth
- RAID level (mirroring)
- Intelligent Data Placement (short stroking)
- O/S and database block size
- ASM vs. File System

Striping refers to distributing files across multiple hard drives to enable parallelized access and to maximize IOPS. A Pure Storage array consists of 22 solid state disks per shelf, and the Purity Operating Environment automatically distributes data across all drives in the array using an algorithm designed to optimize performance and provide redundancy. In other words, the striping is automatic.

The Pure Storage redundancy technology is called RAID-3D, and it is designed specifically to protect against the 3 failure modes specific to flash storage: device failure, bit errors, and performance variability. No other form of RAID protection is needed so you don't need to compromise capacity or performance for data protection. RAID is automatic.

Just as striping and mirroring are irrelevant on a Pure Storage array, so is block size. Pure Storage is based on a fine-grained 512-byte geometry, so there are no block alignment issues as you might encounter in arrays designed with, for example, a 4KB geometry. Another benefit is a substantially higher deduplication rate than seen on other arrays offering data reduction.

Other flash vendors have architected their solution around the new Advanced Format (AF) Technology which allows for 4KB physical sector sizes instead of the traditional 512B sector size. But since solid-state disks don't have sectors or cylinders or spindles, The Purity Operating Environment is designed from the ground up to take advantage of flash's unique capabilities. The benefit to the user is that you can realize the benefits of flash performance without being shackled to any of the constraints of the disk paradigm.

In this document we provide information to help you to optimize the Pure Storage FlashArray™ for your Oracle database workload. Please note that these are general guidelines that are appropriate for many workloads, but as with all guidelines you should verify that they are appropriate for your specific environment.

Operating System Recommendations

Pure Storage has operating system recommendations that apply to all deployments: databases, VDI, etc. These recommendations apply whether you are using Oracle Automatic Storage Management (ASM), raw devices or a file system for your database storage.

Multipath Configuration

Always use multipathing on a Pure Storage FlashArray™. Our all-purpose configuration (tested on RHEL 6.3 and Ubuntu 11.04) is as follows:

/etc/multipath.conf :

```
defaults {
    polling_interval    1
}

devices {
 device {
 vendor "PURE"
 path_selector "round-robin 0"
 path_grouping_policy multibus
 rr_min_io 1
 path_checker tur
 fast_io_fail_tmo 10
 dev_loss_tmo 30
 }
}
```

The settings assigned to “PURE” devices are explained here:

Parameter	Meaning
polling_interval 1	Specifies the frequency (in seconds) to check that each path is alive.
vendor “PURE”	Applies these settings to PURE devices only.
path_selector “round-robin 0”	Loop through every viable path. This setting does not consider path queue length or service time.
path_grouping_policy multibus	Places all paths in a single priority group. This setting ensures that all paths are in use at all times, preventing a latent path problem from going unnoticed.
rr_min_io 1	Sends 1 I/O to each patch before switching to the next path. Using higher numbers can result in inferior performance, as I/O bursts are sent down each path instead of spreading the load evenly.
path_checker tur	“tur” uses the SCSI command Test Unit Ready to determine if a path is working. This is different from the default RHEL setting of readsector0/direction. When a Pure Storage array is failing over read operations will not be serviced, but we should continue to respond to Test Unit Ready. This setting keeps multipath from propagating an I/O error up to the application even beyond the SCSI device timeout.
fast_io_fail_tmo 10	Specifies the number of seconds the SCSI layer will wait after a problem has been detected on a fibre channel port before failing I/O to devices on that remote port. This setting should be less than dev_loss_tmo.
dev_loss_tmo 30	Specifies the number of seconds the SCSI layer will wait after a problem has been detected on a fibre channel remote port before removing it from the system.

SCSI Device Settings

For optimum performance, we also recommend the following device settings.

1. Set the block device scheduler to [noop]. This setting avoids wasting CPU resources on I/O scheduling. The default value is [cfq] (completely fair queuing).
2. Set rq_affinity to 2. This setting avoids interrupt contention on a particular CPU by scheduling I/O on the core that originated the process. The default behavior is to schedule processes on the first core in a package unit which tends to lock cross calls on 1 or 2 cores (equal to the number of sockets in your system).
3. Set add_random to 0. This setting reduces CPU overhead due to entropy collection (for activities such as generating ssh keys).

You can effect these settings either manually (not recommended but ok if LUNS are already in use with different settings) or automatically.

Manually set the scheduler:

```
for disk in `lsscsi | grep PURE | awk '{  print $6 }'`
do
      echo noop > /sys/block/${disk##/dev/}/queue/scheduler
done
```

Manually set rq_affinity:

```
for disk in `lsscsi | grep PURE | awk '{  print $6 }'`
do
      echo 2 > /sys/block/${disk##/dev/}/queue/rq_affinity
done
```

Manually set add_random:

```
for disk in `lsscsi | grep PURE | awk '{  print $6 }'`
do
      echo 0 > /sys/block/${disk##/dev/}/queue/add_random
done
```

We recommend using udev to make these settings automatic across reboots and any other udev triggering event, such as a drive replacement. The recommended rules file is as follows:

```
# Recommended settings for Pure Storage FlashArray.

# Use noop scheduler for high-performance solid-state storage

ACTION=="add|change", SUBSYSTEM=="block", ENV{ID_VENDOR}=="PURE",
ATTR{queue/scheduler}="noop"

# Reduce CPU overhead due to entropy collection

ACTION=="add|change", SUBSYSTEM=="block", ENV{ID_VENDOR}=="PURE",
ATTR{queue/add_random}="0"
```

```
# Schedule I/O on the core that initiated the process

ACTION=="add|change", SUBSYSTEM=="block", ENV{ID_VENDOR}=="PURE",
ATTR{queue/rq_affinity}="2"
```

Save this file as

- `/etc/udev/rules.d/99-pure-storage.rules` [RHEL 6.x]
- `/lib/udev/rules.d/99-pure-storage-rules` [Ubuntu]

You can activate the udev rules with:

```
udevadm trigger
```

Process Prioritization and Pinning

The log writer (ora_lgwr_{ORACLE_SID}) is often a bottleneck for extremely heavy OLTP workloads since it is a single process and it must persist every transaction. A typical AWR Top 5 Timed Foreground Events report might look like this:

Top 5 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
enq: HW - contention	2,361	56,081	23753	65.84	Configuration
log file sync	2,781	23,018	8277	27.03	Commit
enq: TX - row lock contention	36,708	2,973	81	3.49	Application
buffer busy waits	9,338	1,969	211	2.31	Concurrency
DB CPU		1,208		1.42	

If your AWR report shows high waits on LOG_FILE_SYNC or LOG_FILE_PARALLEL_WRITE, you can consider making these adjustments. However, do not do so unless your system has eight or more cores.

To increase log writer process priority:

- Use renice
- E.G. if log writer process id is 27250: `renice -n 20 27250`
- Probably not adviseable if your system has few than eight cores

To pin log writer to a given core:

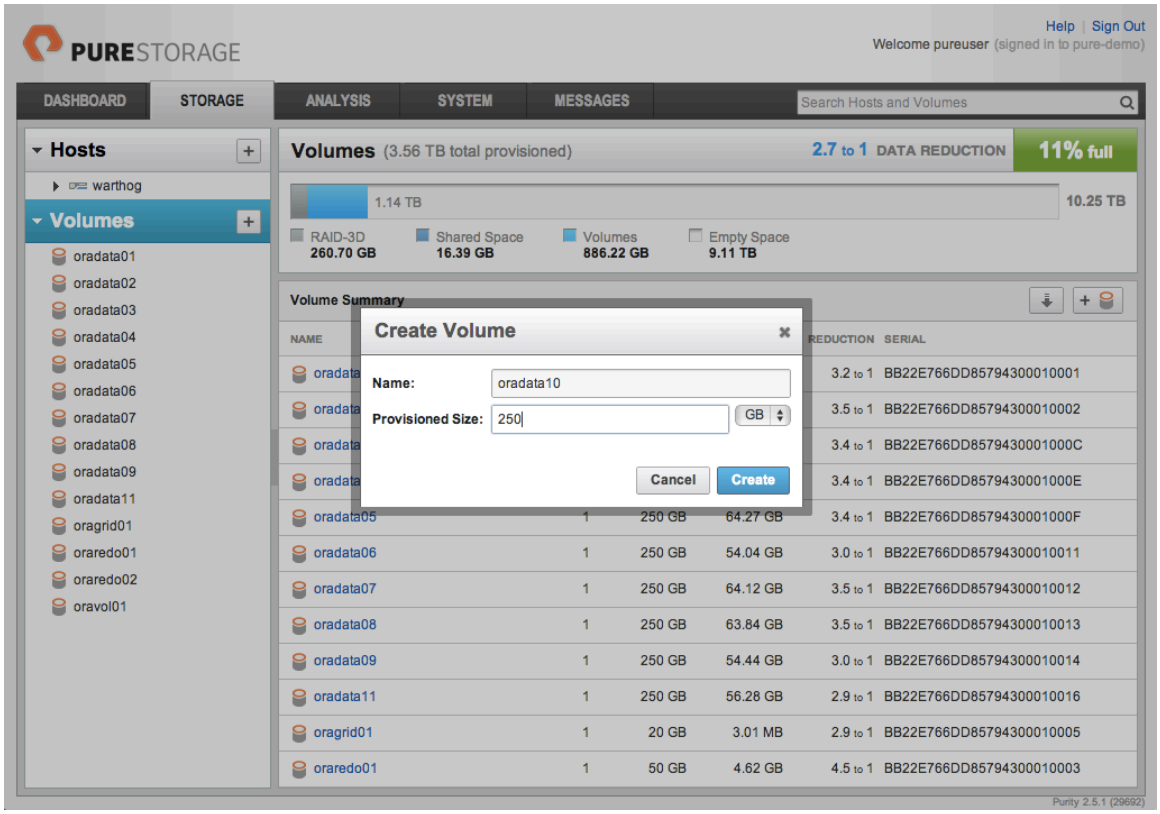
- Use taskset
- E.G. if log writer process id is 27250: `taskset -p 1 27250`
- Probably not adviseable if you have fewer than 12 cores

We have found that the Pure Storage FlashArray™ can sustain redo log write rates over 100MB/s.

Provisioning Storage on the Pure Storage FlashArray™

You can provision storage from either the Pure Storage GUI management tool, or the command line, as illustrated here.

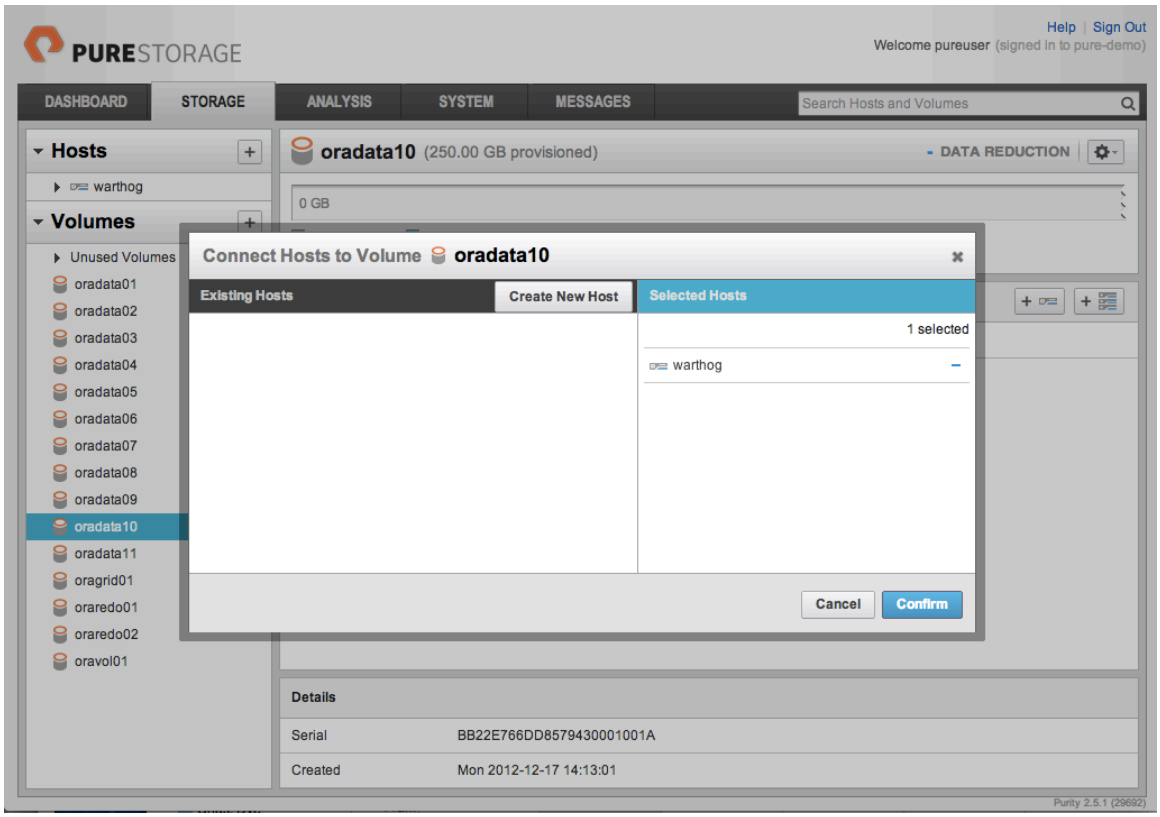
1. Create a volume using either the GUI or CLI



Command line equivalent:

```
purevol create -size 250G oravol10
```


2. Connect the volume to the host



Command line equivalent:

```
purevol connect --host warthog oradata10
```

3. Scan the new volume on the database server (as root):

```
# rescan-scsi-bus.sh -i -r
```

4. Flush any unused multipath device maps (as root):

```
# multipath -F
```


5. Detect and map the new volume with multipath (as root):

```
# multipath -v2
```

Note the new volume's unique device identifier (UUID) which is the same as the Serial number seen in the Details section of the GUI. In this case it's `3624a9370bb22e766dd8579430001001a`

```
[root@warthog ~]# multipath -v2
Dec 17 14:20:01 | mpatha: ignoring map
create: mpathbk (3624a9370bb22e766dd8579430001001a) undef PURE,FlashArray
size=250G features='0' hwhandler='0' wp=undef
`-+- policy='round-robin 0' prio=1 status=undef
  |- 1:0:0:4   sdk  8:160  undef ready running
  |- 2:0:0:4   sdg  8:96   undef ready running
  |- 3:0:0:4   sds  65:32  undef ready running
  |- 4:0:0:4   sdai 66:32  undef ready running
  |- 2:0:1:4   sdbn 68:16  undef ready running
  |- 4:0:1:4   sdcg 69:64  undef ready running
  |- 1:0:1:4   sdch 69:80  undef ready running
  `- 3:0:1:4   sddn 71:80  undef ready running
```

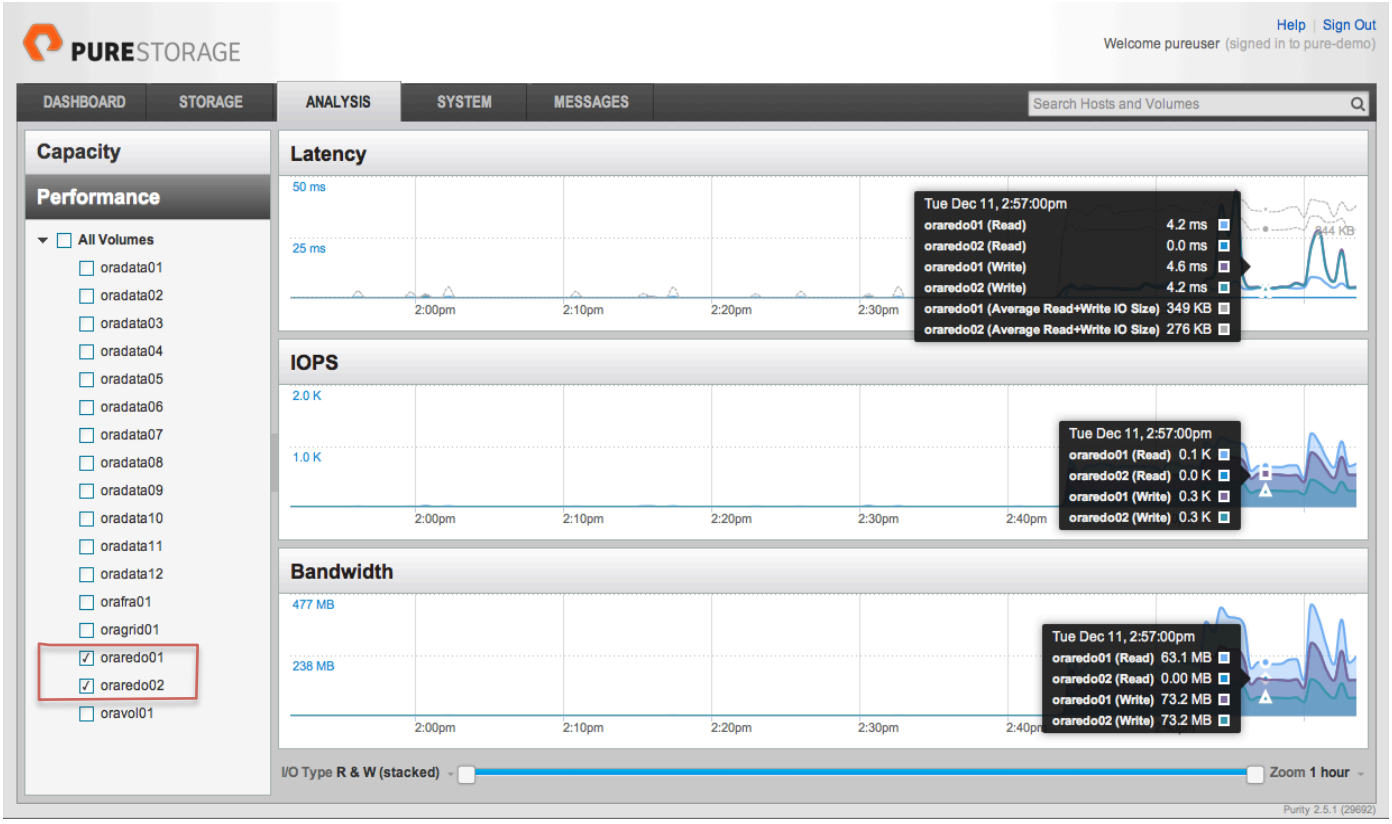
At this point, the newly provisioned storage is ready to use. You can either create a file system on it, or use it as an Automatic Storage Management (ASM) disk.

ASM versus File System

On a Pure Storage FlashArray™, there is no significant performance benefit to using ASM over a traditional file system, so the decision can be driven by your operational policies and guidelines. Whichever storage mechanism you choose will perform well. From a DBA's perspective, ASM does offer additional flexibility not found with file systems, such as the ability to move ASM disks from one disk group to another, resizing disks, and adding volumes dynamically.

Recommendations Common to ASM and File System

Unlike traditional storage, IOPS are not a function of LUN count. In other words, you get the same IOPS capacity with 1 LUN as you do with 100. However, since it is often convenient to monitor database performance by I/O type (for example, LGWR, DBWR, TEMP), we recommend creating ASM disk groups or file systems dedicated to these individual work loads. This strategy allows you to observe the characteristics of each I/O type either at the command line with tools like iostat and pureadm, or with the Pure Storage GUI.



In addition to isolating I/O types to individual disk groups, you should also locate the flash recovery area (FRA). We also recommend opting for a few large LUN's per disk group.

If performance is a critical concern, we also recommend that you do not multiplex redo logs. It is not necessary for redundancy since RAID-3D provides protection against media failures. Multiplexing redo logs introduces a performance impact of up to approximately 10% for a heavy OLTP workload. If your operations policy requires you to multiplex redo logs, we recommend placing the group members in

separate disk groups or file systems. For example, you can create 2 disk groups, REDOSSD1 and REDOSSD2 and multiplex across them:

```
alter database add logfile group 1 ('+REDOSSD1', '+REDOSSD2' ) size 2g;
```

```
SELECT     g.name groupname, d.path, g.sector_size, g.block_size, g.state
FROM       v$asm_diskgroup g, v$asm_disk d
WHERE      d.group_number = g.group_number
AND        g.name LIKE '%REDOSSD%'
ORDER BY   g.name;

                            Sector Block
Group Name Path                Size  Size STATE
---------- --------------- ------ ----- -----------
REDOSSD1   /dev/dm-3             512  4096 CONNECTED
REDOSSD2   /dev/dm-28            512  4096 CONNECTED

2 rows selected.
```

Finally, while some flash storage vendors recommend a 4K block size for both redo logs and the database itself (to avoid block misalignment issues), Pure Storage does not. Since the Pure Storage FlashArray™ is designed on a 512 byte geometry, we never have block alignment issues. Performance is completely independent of block size.

ASM Specific Recommendations

In Oracle 11gR3 the default striping for ONLINELOG template changed from FINE to COARSE. In OLTP workload testing we found that the COARSE setting for redo logs performs about 20% better. Since the Pure Storage FlashArray™ includes RAID-3D protection, you can safely use External Redundancy for ASM diskgroups. Other factors such as sector size and AU size do not have a significant bearing on performance.

ASM Disk Group Recommendations

Disk Group	Sector Size	Strip	AU Size	Redundancy	Notes
ORACRS	512	COARSE	1048576	External	Small disk group for CRS
ORADATA	512	COARSE	1048576	External	Database segments
ORAREDO	512	COARSE	1048576	External	Redo logs
ORAFRA	512	COARSE	1048576	External	Flash Recovery Area

ASM Space Reclamation

As you drop, truncate or resize database objects in an ASM environment, the space metrics reported by the data dictionary (DBA_FREE_SPACE, V$ASM_DISKGROUP, V$DATAFILE, etc.) will reflect your changes as expected. However, these actions may not always trim (free) space on the array immediately.

Oracle provides a utility called ASM Storage Reclamation Utility (ASRU) which expedites the trim operation. For example, after dropping 1.4TB of tablespaces and data files, Oracle reports the newly available space in V$ASM_DISKGROUP, but `puredb list space` still considers the space to be allocated. Consider the case when we drop the 190GB tablespace ASRUDEMO which is in the ORADATA disk group:

Before dropping the tablespace:

```
SELECT      name,
            total_mb/(1024) total_gig,
            free_mb/(1024) free_gig,
            (total_mb-free_mb)/1024 used_gig
FROM        v$asm_diskgroup;
```

```
                              Group          Group       Group
                              Total           Free        Used
NAME                             GB             GB          GB
-------------------- ------------- -------------- -----------
ORADATA                   2,250.00         388.50    1,861.50
ORAFRA                    1,643.03         435.77    1,207.25
ORAGRID                      20.00          19.94         .06
REDOSSD2                     50.00          49.94         .06
REDOSSD                      50.00           9.89       40.11

5 rows selected.
```

And on the storage array:

```
pureuser@pure-demo> purevol list --space
Name          Size    Data Reduction  System   Shared Space  Volume   Snapshots  Total
oradata01     250G    1.6 to 1        659.97M  -             69.0G    0.00       69.66G
oradata02     250G    1.5 to 1        565.25M  -             116.17G  0.00       116.74G
oradata03     250G    1.5 to 1        498.90M  -             114.71G  0.00       115.21G
oradata04     100G    2.5 to 1        318.67M  -             115.10G  0.00       115.42G
oradata05     100G    2.6 to 1        308.97M  -             114.64G  0.00       114.95G
oradata06     250G    1.8 to 1        995.37M  -             113.18G  0.00       114.18G
oradata07     250G    2.7 to 1        1.15G    -             113.78G  0.00       114.93G
oradata08     250G    3.5 to 1        2.55G    -             111.91   0.00       114.46G
oradata09     250G    2.1 to 1        1.22G    -             113.77G  0.00       114.99G
oradata10     500G    2.9 to 1        3.79G    -             126.49G  0.00       130.28G
```


After we drop the ASRUDEMO tablespace, `v$asm_diskgroup` updates the available space as expected:

```
drop tablespace tpchtab including contents and datafiles;

Tablespace dropped.

SELECT      name,
            total_mb/(1024) total_gig,
            free_mb/(1024) free_gig,
            total_mb-free_mb)/1024 used_gig
FROM        v$asm_diskgroup;

                              Group          Group        Group
                              Total           Free         Used
NAME                             GB             GB           GB
------------------ ------------- ------------- -----------
ORADATA                   2,250.00      1,779.34       470.66
ORAFRA                    1,643.03        435.77     1,207.25
ORAGRID                      20.00         19.94          .06
REDOSSD2                     50.00         49.94          .06
REDOSSD                      50.00          9.89        40.11

5 rows selected.
```

However, we don't see the space recovered on the storage array:

```
pureuser@pure-demo> purevol list --space
Name           Size    Data Reduction  System   Shared Space  Volume   Snapshots  Total
oradata01      250G    1.6 to 1        659.97M  -             69.0G    0.00       69.66G
oradata02      250G    1.5 to 1        565.25M  -             116.17G  0.00       116.74G
oradata03      250G    1.5 to 1        498.90M  -             114.71G  0.00       115.21G
oradata04      100G    2.5 to 1        318.67M  -             115.10G  0.00       115.42G
oradata05      100G    2.6 to 1        308.97M  -             114.64G  0.00       114.95G
oradata06      250G    1.8 to 1        995.37M  -             113.18G  0.00       114.18G
oradata07      250G    2.7 to 1        1.15G    -             113.78G  0.00       114.93G
oradata08      250G    3.5 to 1        2.55G    -             111.91   0.00       114.46G
oradata09      250G    2.1 to 1        1.22G    -             113.77G  0.00       114.99G
oradata10      500G    2.9 to 1        3.79G    -             126.49G  0.00       130.28G
```


Although the array's garbage collection will free the space eventually, we can use the ASRU utility (under the "grid" O/S account on the database server) to trim the space immediately:

```
11:12:39 [asru] grid@warthog 764$ ./ASRU oradata
Checking the system ...done
Calculating the sizes of the disks ...done
Writing the data to a file ...done
Resizing the disks...done
Calculating the sizes of the disks ...done

/u01/app/oracle/product/11.2.0.3/grid/perl/bin/perl -I /u01/app/oracle/product/11.2.0.3/grid/perl/lib/5.10.0 /home/g
id/asru/zerofill 1 /dev/dm-2 66943 256000 /dev/dm-5 66940 256000 /dev/dm-6 66939 256000 /dev/dm-9 66941 256000 /dev/
m-3 66939 256000 /dev/dm-12 66941 256000 /dev/dm-8 66940 256000 /dev/dm-1 66939 256000 /dev/dm-11 66940 256000
189057+0 records in
189057+0 records out
198240632832 bytes (198 GB) copied, 461.619 s, 429 MB/s
189060+0 records in
189060+0 records out
198243778560 bytes (198 GB) copied, 482.291 s, 411 MB/s
189061+0 records in
189061+0 records out
198244827136 bytes (198 GB) copied, 467.587 s, 424 MB/s
189059+0 records in
189059+0 records out
198242729984 bytes (198 GB) copied, 468.869 s, 423 MB/s
189061+0 records in
189061+0 records out
198244827136 bytes (198 GB) copied, 471.956 s, 420 MB/s
189059+0 records in
189059+0 records out
198242729984 bytes (198 GB) copied, 1259.6 s, 157 MB/s
189060+0 records in
189060+0 records out
198243778560 bytes (198 GB) copied, 205.757 s, 963 MB/s
189061+0 records in
189061+0 records out
198244827136 bytes (198 GB) copied, 457.291 s, 434 MB/s
189060+0 records in
189060+0 records out
198243778560 bytes (198 GB) copied, 473.093 s, 419 MB/s

Calculating the sizes of the disks ...done
Resizing the disks...done
Calculating the sizes of the disks ...done
Dropping the file ...done
```

After the ASRU run, the recovered space is visible in the Physical Space column of the `puredb list space` report:

```
pureuser@pure-demo> purevol list --space
Name            Size    Data Reduction  System   Shared Space  Volume   Snapshots  Total
oradata01       250G    1.6 to 1        106.38M  -             12.45G   0.00       12.55G
oradata02       250G    1.5 to 1        2.60M    -             28.51G   0.00       28.51G
oradata03       250G    1.5 to 1        2.95M    -             27.82G   0.00       27.82G
oradata04       100G    2.5 to 1        3.11M    -             28.86g   0.00       28.86G
oradata05       100G    2.6 to 1        3.02M    -             28.66G   0.00       28.66G
oradata06       250G    1.8 to 1        3.64M    -             28.17G   0.00       28.17G
oradata07       250G    2.7 to 1        19.92M   -             30.30G   0.00       30.31G
oradata08       250G    3.5 to 1        68.96M   -             48.32G   0.00       48.39G
oradata09       250G    2.1 to 1        3.58M    -             29.33G   0.00       29.33G
oradata10       500G    2.9 to 1        3.72M    -             32.81G   0.00       32.81G
```

ASMLib and Alternatives

ASMLib is an Oracle-provided utility that allows you to configure block devices for use with ASM. Specifically, it marks devices as ASM disks, and sets their permissions so that the o/s account that runs ASM (typically either `grid` or `oracle`) can manipulate these devices. For example, to create an ASM disk named MYASMDISK backed by `/dev/dm-2` you would issue the command:

```
/etc/init.d/oracleasm createdisk MYASMDISK /dev/dm-2
```

Afterwards, /dev/dm-2 would appear still have the same ownership and permissions, but ASMLib will create a file `/dev/oracleasm/disks/MYASMDISK` owned by the O/S user and group the is identified in `/etc/sysconfig/oracleasm`. You tell the ASM instance to look for potential disks in this directory though the `asm_diskstring` initialization parameter.

The problem with ASMLib is that Oracle stopped providing it with RHEL 6.3 and beyond, although it is still provided for Oracle Unbreakable Linux. Fortunately, there are simpler alternatives that work on RHEL (as well as Oracle Unbreakable Linux).

ASMLib Alternative 1: udev

On RHEL 6.x you can use udev to present devices to ASM. Consider the device mpathbk created above. The device is created as `/dev/mapper/mpathbk`, linked to `/dev/dm-4` and owned by root:

```
[root@warthog ~]# ls -l /dev/mapper/mpathbk
lrwxrwxrwx 1 root root 7 Dec 17 15:32 /dev/mapper/mpathbk -> ../dm-4
[root@warthog ~]# ls -l /dev/dm-4
brw-rw---- 1 root root 253, 4 Dec 17 15:32 /dev/dm-4
```

Perform the following steps to change the device ownership to grid:asadmin.

1. Create an entry for the device in the udev rules file `/etc/udev/rules.d/12-dm-permissions.rules` as follows

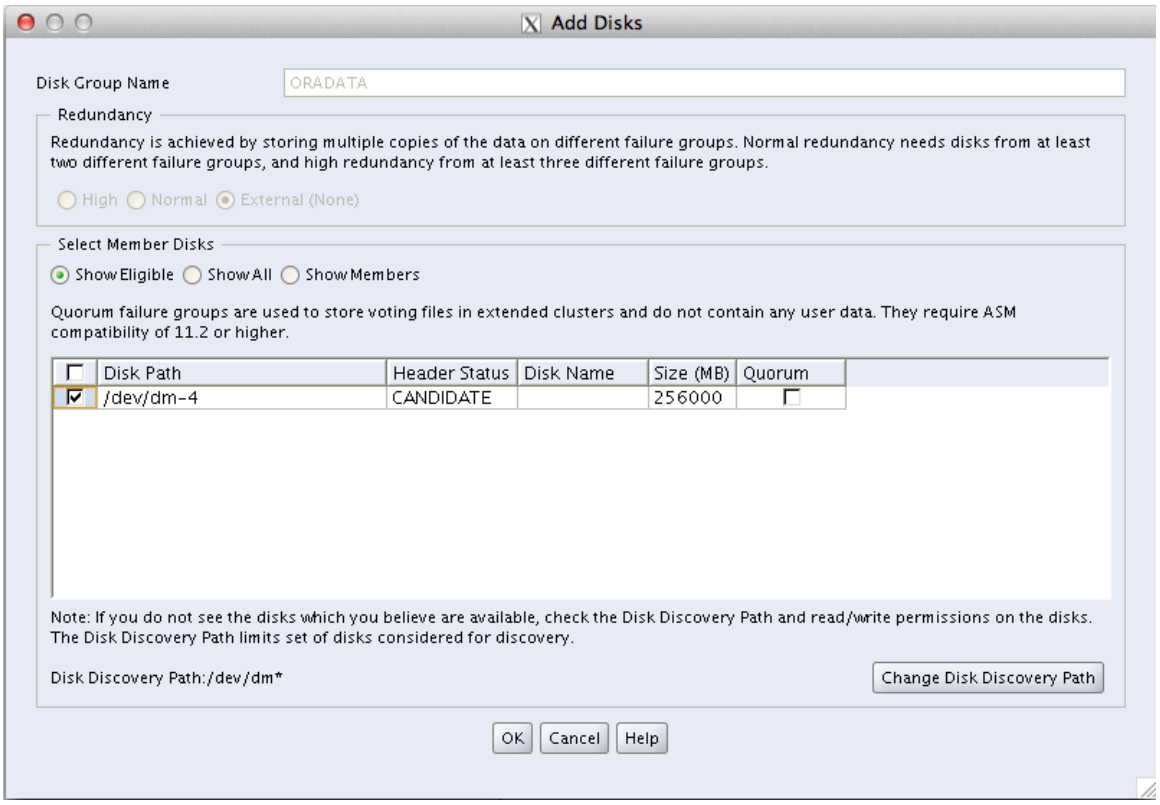
```
# oradata10
ENV{DM_UUID}=="mpath-3624a9370bb22e766dd8579430001001a", OWNER:="grid", GROUP:="asmadmin", MODE:="660"
```

2. Use udevadm to trigger udev events and confirm the change in ownership for the block device:

```
[root@warthog ~]# udevadm trigger
[root@warthog ~]# ls -l /dev/dm-4
brw-rw---- 1 grid asmadmin 253, 4 Dec 17 15:33 /dev/dm-4
```

Note the change in ownership to grid:asmadmin.

3. Use sqlplus or asmca to create a new disk group or to put the new device in an existing disk group. Since the Purity Operating Environment provides RAID-3D you can safely use External Redundancy for the disk group. Note that your `asm_diskstring` (discovery path) should be /dev/dm*



After clicking “OK” the device will be added to the disk group and an ASM rebalance operation will execute automatically. We recommend using the same size disk for all members of a disk group for ease of rebalancing operations.

ASMLib Alternative 2: multipath

The udev rules described above do not work on Linux 5.7, but you can effect the ownerships required for ASM through the mulitpaths stanza of the `/etc/multipath.conf` file:

```
defaults {
    polling_interval 1
}
devices {
 device {
 vendor "PURE"
 path_selector "round-robin 0"
 path_grouping_policy multibus
 rr_min_io 1
 path_checker tur
 fast_io_fail_tmo 10
 dev_loss_tmo 30
 }
}
 multipaths {
  multipath {
  wwid 3624a937034fedb8251d5dcae0001028e
  alias ASMDISK01
  uid grid
  gid asmadmin
  mode 660
  }
  multipath {
  wwid 3624a937034fedb8251d5dcae0001028f
  alias ASMDISK02
  uid grid
  gid asmadmin
  mode 660
  }
  multipath {
  wwid 3624a937034fedb8251d5dcae00010290
  alias ASMDISK03
  uid grid
  gid asmadmin
  mode 660
  }

 }
```

These entries will create devices with the proper ownership and permissions in `/dev/mapper`. In order for ASM to recognize these devices, the `asm_diskstring` should be set to `/dev/mapper`.

File System Recommendations

There is no significant performance penalty for using a file system instead of ASM. As with ASM, we recommend placing data, redo, and the flash recovery area (FRA) onto separate volumes for ease of administration. We also recommend using the ext4 file system and mount it with discard and noatime options. Below is a sample `/etc/fstab` file showing mount points /u01 (for oracle binaries, trace files, etc.), `/oradata` (for datafiles) and `/oraredo` (for online redo logs).

```
#
# /etc/fstab
# Created by anaconda on Tue Sep 25 16:08:44 2012
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=149604f7-a6f1-4cf8-b7f8-ca90d7eec668 /                       ext4    defaults        1 1
UUID=f039ece7-701c-4a05-ad6b-7d9d7825cee8 /boot                   ext4    defaults        1 2
UUID=c08a6ce8-2460-4307-b9a1-13d74e2db9c5 swap                    swap    defaults        0 0
UUID=42ddf292-c500-4c22-85e3-dbd83a74c50c /u01                    ext4    discard,noatime 1 2
UUID=be1eb882-2dc2-4cdb-a831-02e935f5abf8 /oradata                ext4    discard,noatime 1 2
UUID=58b85ef0-2371-4185-9cfa-c4e002c83b25 /oraredo                ext4    discard,noatime 1 2
tmpfs                   /dev/shm                tmpfs   defaults        0 0
devpts                  /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                   /sys                    sysfs   defaults        0 0
proc                    /proc                   proc    defaults        0 0
```

The man page for mount describes the /discard flag as follows:

```
discard/nodiscard
Controls whether ext4 should issue discard/TRIM commands to the underlying
block device when blocks are  freed.  This is useful for SSD devices and
sparse/thinly-provisioned LUNs, but it is off by default until sufficient
testing has been done.
```

Mounting the ext4 file system with the discard flag causes freed space to be trimmed immediately, just as the ASRU utility trims the storage behind ASM disk groups.

Oracle Settings

For the most part, you don't need to make changes to your Oracle configuration in order to realize immediate performance benefits on a Pure Storage FlashArray™. However, if you have an extremely heavy OLTP workload, there are a few tweaks you can make that will help you squeeze the most I/O out of your system. In our testing, we found that the following settings increased performance by about 5%

init.ora settings

```
_high_priority_processes='LMS*|LGWR|PMON'
```

- Sets processes scheduling priority to RR
- Minimizes need to "wake" LGWR
- Underscore parameter: consult oracle support

```
filesystemio_options = SETALL
```

- Allows asynch i/o

```
log_buffer = {at least 15MB}
```

- Values over 100MB are not uncommon

Use the CALIBRATE_IO Utility

Oracle provides a built in package dbms_resource_manager.calibrate_io which, like the ORION tool, generates workload on the I/O subsystem. However, unlike ORION, it works with your running Oracle database, and it generates statistics for the optimizer. Therefore, you should run calibrate_io and gather statistics for your application schema at least once before launching your application.

The calibrate_io script as provided in the Oracle documentation and presented here using our recommended values for <DISKS> and <MAX_LATENCY>.

```
SET SERVEROUTPUT ON
DECLARE
  lat  INTEGER;
  iops INTEGER;
  mbps INTEGER;
BEGIN
-- DBMS_RESOURCE_MANAGER.CALIBRATE_IO (<DISKS>, <MAX_LATENCY>, iops, mbps, lat);
   DBMS_RESOURCE_MANAGER.CALIBRATE_IO (1000, 10, iops, mbps, lat);

  DBMS_OUTPUT.PUT_LINE ('max_iops = ' || iops);
  DBMS_OUTPUT.PUT_LINE ('latency  = ' || lat);
  dbms_output.put_line('max_mbps = ' || mbps);
end;
/
```

Typically you will see output similar to:

```
max_iops = 134079
latency  = 0
max_mbps = 1516
```


Conclusion

Many of the traditional architecture decisions and compromises you have had to make with traditional storage are not relevant on a Pure Storage FlashArray™. You do not need to sacrifice performance to gain resiliency, nor do you need to change existing policies that you may already have in place. In other words, there is no wrong way to deploy Oracle on Pure Storage; you can expect performance benefits out of the box.

That said, there are some configuration choices you can make to increase flexibility and maximize performance:

- Use the Pure Storage recommended multipath.conf settings
- Set the scheduler, rq_affinity, and entropy for the Pure Storage devices
- Separate differate I/O work loads to dedicated LUNS for enhanced visibility
- If you use a file system for data files, use ext4
- Always run calibrate_io

Feel free to contact Chas. Dye, Database Solutions Architect (cdye@purestorage.com) if you have questions or if you would like to discuss the suitability of Pure Storage in you environment.