# AIX Mirror Pools

Basic LVM mirroring has been available in AIX for years and I think it is great. The added value of mirror pools should come from the ease in maintaining for example 2 copies across 2 remote sites. It should help in making absolutely sure there is a complete copy on each site and help preventing any misconfiguration. If “super strictness” is specified and the disks are assigned to the correct pool, it should not be possible to end up in a situation where mirroring is not protecting data against (single) disk or site failure. For that, everything needs to be mirrored and there needs to be a copy in each pool. Currently Mirror Pools are not doing that. Not only is it possible to end up in violating situations, the LVM commands don’t behave in a consistent manner.

A PMR we opened to address our concerns got in the end simply closed with the message “working as designed”.

I find this subject hard to explain in only words, so I’d like to use the following sequence of commands as example:

```
01:    # mkvg -y mirrorpoolvg -S hdisk1 hdisk2
02:    mirrorpoolvg
03:    # mklv -y lv1 mirrorpoolvg 10 hdisk1
04:    lv1
05:    # mklv -y lv2 mirrorpoolvg 10 hdisk1
06:    lv2
07:    # mklv -y lv3 mirrorpoolvg 10 hdisk1
08:    lv3
09:    # chvg -M s mirrorpoolvg
10:    # chpv -p pool1 hdisk1
11:    # chpv -p pool2 hdisk2
12:    # lsvg -P mirrorpoolvg
13:    # chlv -m copy1=pool1 -m copy2=pool2 lv1
14:    # chlv -m copy1=pool1 -m copy2=pool2 lv2
15:    # chlv -m copy1=pool1 -m copy2=pool2 lv3
16:    # mirrorvg -p copy1=pool1 -p copy2=pool2 mirrorpoolvg
17:    0516-1804 chvg: The quorum change takes effect immediately.
18:    # lsvg -P mirrorpoolvg
19:    Physical Volume   Mirror Pool
20:    hdisk1            pool1
21:    hdisk2            pool2
22:    # lsvg -m mirrorpoolvg
23:    Logical Volume    Copy 1            Copy 2            Copy 3
24:    lv1               pool1             pool2             None
25:    lv2               pool1             pool2             None
26:    lv3               pool1             pool2             None
27:    # lslv lv1 |grep POOL
28:    COPY 1 MIRROR POOL: pool1
29:    COPY 2 MIRROR POOL: pool2
30:    COPY 3 MIRROR POOL: None
31:    # readvgda hdisk1 |grep -E "LV |pool\["|grep -v ===
32:    ------- LV 1 ------
33:    mirror_pool[]:  1 2 0
34:    ------- LV 2 ------
35:    mirror_pool[]:  1 2 0
36:    ------- LV 3 ------
37:    mirror_pool[]:  1 2 0
```

```
38:    # chpv -p poolA hdisk1
39:    0516-1812 lchangepv: Warning, existing allocation violates mirror pools.
40:        Consider reorganizing the logical volume to bring it into compliance.
41:    # lsvg -P mirrorpoolvg
42:    Physical Volume   Mirror Pool
43:    hdisk1            poolA
44:    hdisk2            pool2
45:    # lsvg -m mirrorpoolvg
46:    Logical Volume    Copy 1            Copy 2            Copy 3
47:    lv1                                 pool2             None
48:    lv2                                 pool2             None
49:    lv3                                 pool2             None
50:    # lslv lv1 |grep POOL
51:    COPY 1 MIRROR POOL: None
52:    COPY 2 MIRROR POOL: pool2
53:    COPY 3 MIRROR POOL: None
54:    # readvgda hdisk1 |grep -E "LV |pool\["|grep -v ===
55:    ------- LV 1 ------
56:    mirror_pool[]:  1 2 0
57:    ------- LV 2 ------
58:    mirror_pool[]:  1 2 0
59:    ------- LV 3 ------
60:    mirror_pool[]:  1 2 0


61:    # chpv -p poolB hdisk2
62:    0516-1812 lchangepv: Warning, existing allocation violates mirror pools.
63:        Consider reorganizing the logical volume to bring it into compliance.
64:    # lsvg -P mirrorpoolvg
65:    Physical Volume   Mirror Pool
66:    hdisk1            poolA
67:    hdisk2            poolB
68:    # lsvg -m mirrorpoolvg
69:    Logical Volume    Copy 1            Copy 2            Copy 3
70:    lv1               poolB                               None
71:    lv2               poolB                               None
72:    lv3               poolB                               None
73:    # lslv lv1 |grep POOL
74:    COPY 1 MIRROR POOL: poolB
75:    COPY 2 MIRROR POOL: None
76:    COPY 3 MIRROR POOL: None
77:    # readvgda hdisk1 |grep -E "LV |pool\["|grep -v ===
78:    ------- LV 1 ------
79:    mirror_pool[]:  1 2 0
80:    ------- LV 2 ------
81:    mirror_pool[]:  1 2 0
82:    ------- LV 3 ------
83:    mirror_pool[]:  1 2 0
```

## Part 1: lines 01->37

In this first part, a super strict VG with 2 disks and 3 LVs gets created. The LVs are mirrored across 2 mirror pools.
I guess you could say: "So far, so good".

Although not a real problem, I would have preferred another approach here: The volume group is made "super strict" at line 9, while the volume group is not at all "super strict" yet. At that point nothing is mirrored or protected. It is only after the mirrorvg at line 16 is completed that the volume group is protected and it deserves the label "super strict". Before line 16 nothing is actually mirrored and no warning messages are given.
What I would have preferred is that you would have to bring the VG in a "super strict" state first, before you could do the `chvg -M s mirrorpoolvg`. Once the VG is labeled "super strict", actions that would break the super strictness are no longer allowed. If for some reason such actions would be needed, you would have to turn off the super strictness first.
In such a manner *super strictness* would be really enforced and if `lsvg` would say "strict" it would really mean the config is fine.

## Part 2: lines 38->60

Here 1 disk is moved to another pool and the config is shown using the same commands as in part 1.

`lsvg -p` (lines 41-44) shows the expected result.

The other commands return inconsistent output. In my opinion the only correct output is returned by the `readvgda` output (lines 55-60). Its output is unchanged compared to part 1. This is in my opinion correct, because the lv copy assignment did not change by the `chpv`. I would also expect to see unchanged output for `lsvg -m` (lines 45-49) and `lslv` (lines 50-53). Unfortunately the output from `lslv` at line 51 says "none", just as if the first copy was not assigned to any pool. `lsvg` shows an empty field for copy 1. These commands don't agree!

I asked myself: "what is `lsvg -m` or `'lslv lvx|grep POOL'` really reporting?" (manpage not helping me here):

A. Are they reporting the actual location of copyX for a logical volume? OR
B. Are they reporting what has been specified as target location for each copy of a logical volume?

Reporting the actual allocation can quickly become rather complex. So, I suppose reporting the configured assignment is the option choosen and the way to go. This means the output of `lsvg -m` and `lslv` should not have changed at all. Either way and more importantly the output of both commands should be consistent!

Another point that I'd like to make here is that, again, it is possible to violate the mirror pools. The chpv at line 38 breaks the super-strictness. One could argue that there is a warning message, but this warning is not repeated or mentioned elsewhere (AFAIK). If this warning is missed (e.g. when part of an automated provisioning), you may get the impression that your data is protected by the mirror pools while they may not be. `lsvg -m`, `lslv` and `lsvg -P` continue to give the impression everything is fine.

## Part 3 (lines 61-83)

Here the second disk is also moved to another pool.
It is more or less the same thing as in part 2, but it just takes this strange logic a step further. Not only is `lsvg -m` and `lslv` output still inconsistent, now the first copy of each lv seems to be assigned to poolB!

Where did that come from? You could say that copy 2 was assigned to pool 2 and pool 2 was converted to pool B. So, in some way it may have made sense if copy 2 would be in poolB, but why is it copy 1?

OK, I know by inspecting `readvgda` output how things are implemented and I know where this strange behavior is coming from, but as user I shouldn't have to know. It should "just work". Much like *failure groups* in GPFS "just work". Mirror pools should just work like other LV policies. For inter/intrapolicy you can just say where you prefer your LV to be allocated and a `reorgvg` will attempt to achieve this. It is not, because for example the middle of the disk is already taken by some other LV, that `lslv` will suddenly say that I want my LV to be created on the edge of a disk.

Currently the mirror pool implementation is a bit like the following code sequence where you expect me to accept that the final statement will print "30" just because of the way this has been implemented.

```
A=1
B=2
sum=A+B
A=10
B=20
print sum -> ??
```

In the same way I find it hard to accept that LV pool assignments change when only PV operations take place afterwards.

## Bottom line ...

Having said all that, here are our requests:

- If a VG is to be made "super strict": only allow a clean starting situation before ‘`chvg` -M s‘ can succeed and don’t allow any operation afterwards that would break the rules.

  Alternative: if the above cannot be achieved, then LVM could continue to repeat the warnings (like on line 39) with every LVM command until mirror pool rules are met. Maybe a message could be added to the errpt, which could be used to trigger errornotify actions.
- It would be very useful to have a command to check if mirror policy rules are met and data is properly protected. Not only should the intended configuration be checked, but also the physical allocation, stale PPs, ...
- The mirror pool assignments for each of the copies of a LV should continue to exist even after moving disks between mirror pools. `reorgvg` should then be usable to bring a possibly violating state in the intended state. If what the user requests is not achievable, a warning message should be displayed.
- `lsvg` and `lslv` should show the same output if the pool name is not found and preferably something more meaningful than “<blank>” or “none”.