Brief Advertisement for *pyblk* and RFC

November 9, 2015

Abstract

Storage configurations these days can be large, intricate, and many-layered. When there is a symptom of some problem, the cause of the problem may be hard to figure out. pyblk is an idea for a tool to enhance the ability of the storage administrator to comprehend and visualize a storage configuration and to diagnose and remediate storage problems. It is inspired by the linux utility lsblk, which displays storage configurations in various ways.

pyblk offers some additional features which we expect would prove very useful to the storage administrator. Storage configurations are always DAGS (directed acyclic graphs). lsblk's default display prints the storage graph as a list of trees using a straightforward textual representation. pyblk offers the following significant features beyond those that lsblk offers. First, it is able to write the graph in a structured format that can easily be retrieved and further processed. Second, pyblk graphs can be enriched with more relationships and entities than lsblk acknowledges. Third, pyblk offers a simple diffing capability that automatically calculates and displays differences between storage graphs. Fourth, pyblk offers the ability to display storage graphs in a textual format similar to lsblk and also in a graphical format (which lacks the redundancy of the tree format).

An implementation of pyblk is available at `https://github.com/mulkieran/pyblk/`. Some of the discussion in this document really refers to the PR at `https://github.com/mulkieran/pyblk/pull/2`. If you are interested in trying it out, it's best to check out the PR. See the tox.ini file to get an idea of its dependencies.

lsblk is a handy tool because it offers a readable summary of storage devices and the relationships between them. However, sometimes a storage administrator does not want to know the structure of storage now, but rather the structure of storage at some previous time or whether and how storage may have changed over some period. With its ability to store, retrieve, compare, and display graphs in various ways, pyblk already offers sufficient capabilities to satisfy these needs. We invite suggestions and comments on how to use these existing capabilities and how to integrate them with other tools.

Consider a fairly typical example storage configuration making use of lvm, mdraid, and multipath. For this particular configuration, lsblk generates 188

lines of description in its default format for about 80 devices. This storage configuration is not excessively large or unusually complicated. An example of the first few lines of output is shown below.

```
NAME                                        MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
sda                                           8:0    0 279.4G  0 disk
|-sda1                                       8:1    0   200M  0 part  /boot/efi
|-sda2                                       8:2    0   500M  0 part  /boot
`-sda3                                       8:3    0 278.7G  0 part
  |-rhel_dhcp47--10-root                   253:0    0    50G  0 lvm   /
  |-rhel_dhcp47--10-swap                   253:1    0  11.7G  0 lvm   [SWAP]
  `-rhel_dhcp47--10-home                   253:25   0   217G  0 lvm   /home
sdb                                           8:16   0 931.5G  0 disk
`-WDC_WD10EFRX-68PJCN0_WD-WCC4JKAT9Y7F    253:13   0 931.5G  0 mpath
  `-WDC_WD10EFRX-68PJCN0_WD-WCC4JKAT9Y7F1 253:22   0  15.3G  0 part
    `-md126                                   9:126  0    61G  0 raid5
      `-vg-lv_tdata_corig                  253:29   0    32G  0 lvm
        `-vg-lv_tdata                      253:30   0    32G  0 lvm
          `-vg-lv-tpool                    253:31   0    32G  0 lvm
            |-vg-lv                        253:32   0    32G  0 lvm
            `-vg-thin1                     253:33   0     1T  0 lvm
sdc                                           8:32   0 931.5G  0 disk
`-WDC_WD10EFRX-68PJCN0_WD-WCC4JKECTRRV    253:4    0 931.5G  0 mpath
  `-WDC_WD10EFRX-68PJCN0_WD-WCC4JKECTRRV1 253:21   0  15.3G  0 part
    `-md126                                   9:126  0    61G  0 raid5
      `-vg-lv_tdata_corig                  253:29   0    32G  0 lvm
        `-vg-lv_tdata                      253:30   0    32G  0 lvm
          `-vg-lv-tpool                    253:31   0    32G  0 lvm
            |-vg-lv                        253:32   0    32G  0 lvm
            `-vg-thin1                     253:33   0     1T  0 lvm
...
```

Compare lsblk's output with pyblk's similar output, show below. It has fewer columns and the names are less readable than lsblk's default, but these differences are trivial.

```
NAME                                DEVTYPE
0x600508b1001c79ade5178f0626caaa9c  None
`-/dev/sda                          disk
  |-/dev/sda1                       partition
  |-/dev/sda2                       partition
  `-/dev/sda3                       partition
    |-/dev/dm-0                     disk
    |-/dev/dm-1                     disk
    `-/dev/dm-26                    disk
0xb2590ae94ee25001                  None
|-/dev/sdb                          disk
  `-/dev/dm-6                       disk
    `-/dev/dm-21                    disk
      `-/dev/md126                  disk
        `-/dev/dm-30                disk
          `-/dev/dm-31              disk
            `-/dev/dm-32            disk
              |-/dev/dm-33          disk
              `-/dev/dm-34          disk
0xfb1560374ee25001                  None
|-/dev/sdc                          disk
  `-/dev/dm-10                      disk
    `-/dev/dm-24                    disk
      `-/dev/md126                  disk
        `-/dev/dm-30                disk
          `-/dev/dm-31              disk
            `-/dev/dm-32            disk
              |-/dev/dm-33          disk
              `-/dev/dm-34          disk
...
```

More importantly, observe that pyblk goes further than lsblk in capturing a significant relationship, that of paths with their respective devices. This information is certainly useful to a storage administrator in tracking a particular physical disk[1].

Note that several devices appear more than once. This is a consequence of the fact that the storage configuration is actually a DAG (Directed Acyclic Graph). A representation using trees will have redundant parts wherever one node has more than one parent. For both devices, it is possible to invert the display so that the roots of each tree are the devices at the top of the storage stack and the leaves are the device nodes (in the case of lsblk) or physical devices (typical with pyblk). This does not eliminate the redundancy, it just causes the redundancy to appear elsewhere in the graph.

Now, imagine that there has been some incident or change in behavior, and the storage administrator finds it necessary to investigate. They may be interested in the storage configuration as pyblk would view it now, but they are likely to also be interested in whether a change has occurrred in the storage configuration as a result of, or in connection with the incident or change. Because pyblk is able to store, retrieve, and automatically compare graphs, the storage administrator's job might be made a lot easier. Imagine a facility that automatically caches pyblk graphs over time. These graphs might be cached as a result of a particular event, or regularly, according to a schedule. In any case, these graphs can then be compared. Certain simple differences, for example, new or missing nodes can automatically be computed and the results displayed to the storage administrator.

For example, consider two graphs of storage generated at different times. The first graph requires 215 lines to display, the new one requires 246 lines. Clearly they represent different storage configurations and the textual output is just short eenough that the administrator is likely to discover the differences by a visual inspection of the graph. However, pyblk offers the abiity to do a simple comparison, and the administrator would instantly see the change by deploying its diff tool. Some sample output from the tool is shown below.

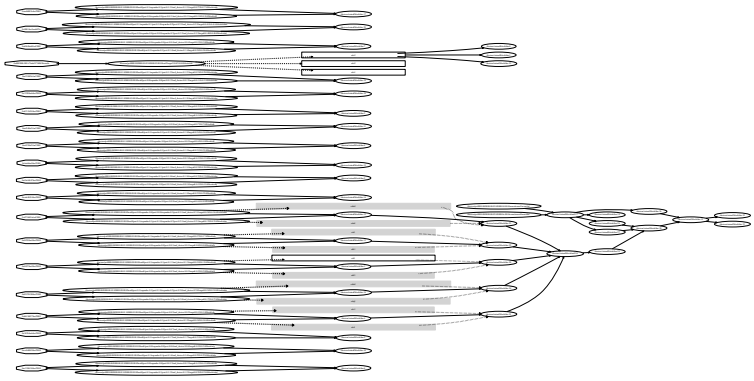
[1] As a consequence of including these additional relationships, pyblk's output is a bit longer than lsblk's, about 215 lines

```
NAME                                  DEVTYPE    DIFFSTATUS
0x600508b1001c79ade5178f0626caaa9c    None       None
`-/dev/sda                            disk       None
  |-/dev/sda1                         partition  None
  |-/dev/sda2                         partition  None
  `-/dev/sda3                         partition  None
    |-/dev/dm-0                       disk       None
    |-/dev/dm-1                       disk       None
    `-/dev/dm-25                      disk       None
0xb2590ae94ee25001                    None       None
|-/dev/sdb                            disk       None
  |-/dev/dm-13                        disk       None
    `-/dev/dm-22                      disk       None
      `-/dev/md126                    disk       None
        `-/dev/dm-29                  disk       None
          `-/dev/dm-30                disk       None
            `-/dev/dm-31              disk       None
              |-/dev/dm-32            disk       None
              `-/dev/dm-33            disk       None
  `+/dev/sdb1                         partition  ADDED
    `+/dev/dm-22                      disk       None
      `-/dev/md126                    disk       None
        `-/dev/dm-29                  disk       None
          `-/dev/dm-30                disk       None
            `-/dev/dm-31              disk       None
              |-/dev/dm-32            disk       None
              `-/dev/dm-33            disk       Non
0xfb1560374ee25001                    None       None
|-/dev/sdc                            disk       None
  |-/dev/dm-4                         disk       None
    `-/dev/dm-21                      disk       None
      `-/dev/md126                    disk       None
        `-/dev/dm-29                  disk       None
          `-/dev/dm-30                disk       None
            `-/dev/dm-31              disk       None
              |-/dev/dm-32            disk       None
              `-/dev/dm-33            disk       None
  `+/dev/sdc1                         partition  ADDED
    `+/dev/dm-21                      disk       None
      `-/dev/md126                    disk       None
        `-/dev/dm-29                  disk       None
          `-/dev/dm-30                disk       None
            `-/dev/dm-31              disk       None
              |-/dev/dm-32            disk       None
              `-/dev/dm-33            disk       None
...
```

The word "ADDED" in the diffstatus column indicates that the device has been added to the graph. What has happened, it turns out, is that `parted --list` has been run, with the result that the partition device nodes, previously removed by multipath udev rules, have been re-created by parted.

While the textual representation of a storage graph is valuable, an image can sometimes be more informative, allowing the administrator to see the overall structure of moderately sized storage configurations at a glance. Consider the image below, which has been aggressively scaled to 6% of its original size. It's not possible to see much detail, but it is possible to see the overall structure more easily than with the textual representation. If you know that partition devices are currently shown as rectangles and physical devices as octagons things become a little clearer. This graph is really the graphical version of the difference graph, part of which is shown above. The newly added partitions are shaded, and quite visible. We see at a glance that not quite all of the partitions were

re-added by parted, one already existed.



At present, pyblk is a simple library that computes, stores, compares, and displays storage configurations in various ways. We believe that the ideas it embodies could be useful in a variety of applications to assist the storage administrator in diagnosing and remediating storage problems. We look forward to your suggestions on ways to make use of its capabilities.