Sanlock troubleshooting

Vojtěch Juránek

oVirt storage team

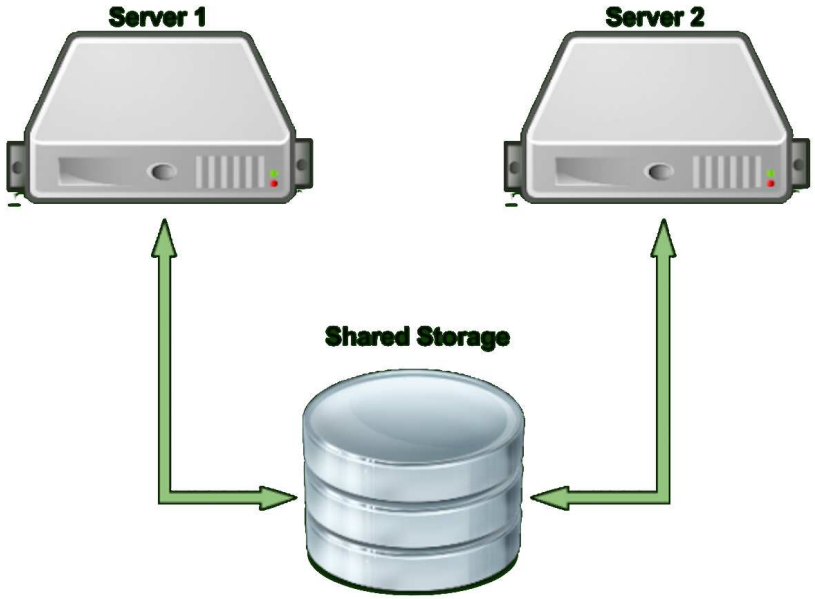
1. 12. 2020, oVirt internal

sanlock troubleshooting: TL&DR

- Check the storage is correctly connected and writeable.
- Check LVs/files `ids`, `leases`, `xleases` are writeable by `sanlock` user.
- Check IO on the given storage domain is not extremely slow.
- Check that the block size of the storage was correctly determined by vdsm.
- Check `/var/log/sanlock.log` for more help.

sanlock

- Distributed lock manager.
- Uses shared storage for exchanging information between participants.
- Can be used to protect any resource, not only data (e.g. keep lease for SPM host in vdsm).



Source: `https://www.linuxjournal.com/content/high-availability-storage-ha-lvm`

sanlock leases

- Δ-leases: used for acquiring/confirming unique ID for each host (only for sanlock internal usage).
- Paxos leases: used for acquiring locks for resources (meaning of the lock is determined by the application).
- Unique host IDs are use for Paxos leases for mapping host to its disk segment.

Δ-Leases

- In sanlock terminology, obtaining the Δ-leases is called to join the *lockspaces*.
- Prevents two hosts to have same ID.
- Lockspace size is 1 MB (for block size 512B).
- Maximum number of hosts is 2000.
- sanlock renews Δ-leases every 20 seconds by writing a new time stamp into appropriate Δ-lease block.
- Lease renewal is used also for checking if host is alive.

Paxos/resource leases

- Fast to acquire.
- In sanlock terminology called *resources*.
- sanlock uses Δ-lease renewals also for Paxos leases renewal in given lockspace.

Lease renewal failure

- Resource lease renewal failure: lease is released automatically by sanlock.
- Δ-lease renewal failure: sanlock will attempt to stop or kill any process (in our case `vdsmd`) holding resource leases in the expiring lockspace.

Default sanlock timeouts

- IO timeout (`io_timeout_seconds`) is configurable, by default 10 s.
- ID renewal 2 * `io_timeout_seconds`, by default 20 s.
- ID renewal fail/timeout (`id_renewal_fail_seconds`) 8 * `io_timeout_seconds`, by default 80 s.

- `watchdog_fire_timeout` is 60 s (hardcoded constant).
- `host_dead_seconds` is by default 140 s (`id_renewal_fail_seconds` + `watchdog_fire_timeout`) - host ID can be acquired by another host by this time.

Usefull commands

- `less /var/log/sanlock.log`
- `sanlock client host_status`
- `sanlock client status`
- `sanlock client log_dump`

sanlock status

```
1 s 78bc1ebc-4f18-4b15-a112-5524fcb198c6:1:/rhev/data-center/mnt/192.168.122.13\:_srv_data_sanlock/78bc1ebc-4f18-4
      b15-a112-5524fcb198c6/dom_md/ids:0
2 r 78bc1ebc-4f18-4b15-a112-5524fcb198c6:SDM:/rhev/data-center/mnt/192.168.122.13\:_srv_data_sanlock/78bc1ebc-4f18-4
      b15-a112-5524fcb198c6/dom_md/leases:1048576:8 p 5578
```

- `s` - lockspace, `r` - resource
- name of lockspace
- local host identifier in lockspace or resource name
- path to storage to use for leases
- offset on path (bytes)
- in case of resource there can be leader version and/or `SH` in case of shared mode

Lockspace/resource options

- `lockspace_name:host_id:path:offset`
- `lockspace_name:resource_name:path:offset`
- `lockspace_name:resource_name:path:offset:leader_version`
- `lockspace_name:resource_name:path:offset:SH`

E.g.

- `sanlock client add_lockspace -s test:1:/dev/loop0:0`
- `sanlock client acquire -r test:res:/dev/loop0:1048576 -p 29755`

sanlock from cmd line: full example

```
1  qemu-img create -f raw /var/tmp/test/sanlock.img 1G
2  losetup --find --show /var/tmp/test/sanlock.img
3  sanlock daemon -w 0
4  sanlock client init -s test:0:/dev/loop0:0
5  sanlock client add_lockspace -s test:1:/dev/loop0:0
6  sanlock client init -r test:res:/dev/loop0:1048576
7  sanlock client command -c /bin/sleep 600 &
8  sanlock client acquire -r test:res:/dev/loop0:1048576 -p
       29755
9  sanlock client release -r test:res:/dev/loop0:1048576 -p
       29755
0  sanlock client rem_lockspace -s test:1:/dev/loop0:0
1  sanlock shutdown
2  losetup -d /dev/loop0
3  rm -f /var/tmp/test/sanlock.img
```

sanlock in vdsm

- Each oVirt host has to join the lockspace (hold Δ-lease) during activation.
- For each SD is one lockspace, lockspace name is SD ID.
- SPM host holds resource lease.
- Host can hold other resource leases (e.g. when running HA VM), but in many cases host doesn't hold and resource lease.
- When there is a problem with the storage, vdsm is not killed by sanlock, unless host/vdsm holds a lease.

sanlock in vdsm metadata

SD metadata related to sanlock:

- `ids` - contains Δ-leases
- `leases` - special/reserved resource leases (lease for SPM)
- `xleases` - user/external leases (e.g. leases for HA VMs). Also contains mapping between lease name and its offset in `leases`. When corrupted, can be rebuild by `vdsm-tool rebuild-xleases`
- Currently `io_timeout_seconds` cannot be change in vdsm, i.e. sanlock uses its default value of 10 s.
- In the future will be configurable, see vdsm io-timeout topic branch

sanlock in vdsm code

Two main modules related to sanlock:

- `clusterlock` module (sanlock initialization, obtaining cluster lock and resource leases)
- `xlease` modules

vdsm udev rule for sanlock

```
1 ENV{DM_LV_NAME}=="ids|leases|xleases", MODE:="0660",
    OWNER:="@VDSMUSER@", GROUP:="@SNLKGROUP@", GOTO="
    lvm_end"
```

Source in `lib/vdsm/storage/vdsm_lvm_rules.template.in`

sanlock troubleshooting: TL&DR

- Check the storage is correctly connected and writeable.
- Check LVs/files `ids`, `leases`, `xleases` are writeable by `sanlock` user.
- Check IO on the given storage domain is not extremely slow.
- Check that the block size of the storage was correctly determined by vdsm.
- Check `/var/log/sanlock.log` for more help.

Resources

- https://pagure.io/sanlock
- Protecting your resources with sanlock (my DevConf 2020 talk)
- E. Gafni, L. Lamport, Disk Paxos
- G. Chockler, D. Malkhi, Light-Weight Leases for Storage-Centric Coordination

Questions?

Thank you!

Questions?