Solaris Volume Manager : Services & Daemons

SVM Services

- Difference between SMF and rc scripts
  - Overview of SMF
  - Overview of rc operation
- List of services
- Location of services
- Core services
- Operational Model
- Relationship between services
- How/where services are started and stopped

SVM 'Services' pre-SMF

- Supported using rc scripts
  - No dependency ordering
  - Difficult to access failure information
- Sxx for start scripts
  - Called with 'start' parameter
- Kyy for stop scripts
  - Called with 'stop' parameter

SVM 'Services' pre-SMF (2)

- 2 rc scripts delivered
  - etc/init.d/svm.sync
  - etc/init.d/svm.init
- Following symbolic links created to those scripts
  - etc/rc0.d/K34svm.sync <-- etc/init.d/svm.sync
  - etc/rc1.d/K34svm.sync <-- etc/init.d/svm.sync
  - etc/rc2.d/S95svm.sync <-- etc/init.d/svm.sync
  - etc/rcS.d/K34svm.sync <-- etc/init.d/svm.sync
  - etc/rcS.d/S35svm.init   <-- etc/init.d/svm.init

SVM Services Under SMF

- SVM Services provided include:
  - /var/svc/manifest/network/rpc
    - meta
    - metamed
    - metamh
    - mdcomm
  - /var/svc/manifest/system
    - metainit
    - mdmonitor

SVM Network Services

- The services required to coordinate disk set administration across computers with shared storage
- RPC
- started/restarted by inetd
- Service Identifier:
  - svc:/network/rpc/<net service name>:default
- Manifest Location:
  - /var/svc/manifest/network/rpc/<net service name>.xml

Network Service Example Manifest

```
<service_bundle type='manifest' name='SUNWmdr:metad'>

<service
        name='network/rpc/meta'
        type='service'
        version='1'>

        <create_default_instance enabled='false' />

        <restarter>
                <service_fmri value='svc:/network/inetd:default' />
        </restarter>

        <dependency name='rpcbind'
            grouping='require_all'
            restart_on='restart'
            type='service'>
                <service_fmri value='svc:/network/rpc/bind' />
        </dependency>
```

SVM System Services

- Services associated with the operation of SVM on the local node
- Service identifier:
  - svc:/system/<system service name>
- Manifest Location;
  - /var/svc/manifest/system/<system service name>.xml
- Method associated with each system service:
  - /lib/svc/method/svc-<system service name>

System Service Example Manifest

```
<service_bundle type='manifest' name='SUNWmdr:metainit'>
<service
    name='system/metainit'
    type='service'
    version='1'>
    <create_default_instance enabled='true' />
    <single_instance />
    <dependency
      name='identity'
      type='service'
      grouping='require_all'
      restart_on='none'>
      <service_fmri value='svc:/system/identity:node' />
    </dependency>

    <dependent
      name='metainit_single-user'
      grouping='optional_all'
      restart_on='none'>
        <service_fmri value='svc:/milestone/single-user' />
    </dependent>
```

System Service Example Manifest (2)

```
   <dependent
          name='metainit-root'
          grouping='optional_all'
          restart_on='none'>
             <service_fmri
 value='svc:/system/filesystem/root' />
       </dependent>
   <exec_method
          type='method'
          name='start'
          exec='/lib/svc/method/svc-metainit'
          timeout_seconds='180' />

       <exec_method
          type='method'
          name='stop'
          exec=':true'
          timeout_seconds='2' />
```

Metainit service method

- Sanity checking
- Runs 'metainit -r' to set up all of the configured metadevices
- Very few changes from init.d script
  - Include of ' /lib/svc/share/smf_include.sh'
  - Use of smf standard error codes

Mdmonitor service method

- Sanity checking
- Runs 'metadevadm -r' on the local set to recompute the pathname and disk specifier from the device id stored in the state database
- Runs 'metadevadm -r -s' for each autotake set
- Run 'metasync -r' to start a resync operation on all devices in need of a resync
- Start the 'mdmonitord' daemon

SVM Service Activation/Deactivation

- Enabling/disabling services is controlled by the SVM applications. Services are only enabled if required and are disabled when no longer needed.
- The smf interface function source is located at usr/src/lib/lvm/libmeta/common/meta_smf.c
- Functions available include:
  - meta_smf_enable
  - meta_smf_disable
  - meta_smf_isonline
  - meta_smf_getmask
- Defines the SVM service classes

SVM Service Classes

- Core
  - mdmonitor
  - metainit
  - meta
- Set
  - metamed
  - metamhd
- MN set
  - mdcomm

Enabling/Disabling Service Classes

- Core
  - Enabled when the first metadb is created in the local set (metadb)
  - Disabled when the last local metadb is deleted (metadb)
- Set
  - Dependency on core services
  - Enabled on all nodes in the diskset when set is created (rpc.metad)
  - Disabled when the last diskset is deleted or the last metadb is deleted (metaset/metadb)
- MN set
  - Dependency on set services
  - Enabled when the MN diskset is created (rpc.metad)
  - Disabled when the last MN diskset is deleted or the last metadb is deleted (metaset/metadb)

SVM Service Debug

- svcs -xv
- This will list the services that are enabled but not running or are preventing another enabled service from running.
- Lists the location of the output file from the service method invocation.

SVM Daemons

- mdmonitord
- rpc.metad
- rpc.metamedd
- rpc.metamhd
- rpc.mdcommd

mdmonitord

- Monitors and checks mirrors, RAID5 metadevices, and hot spares
- Checks devices in the local set and any disksets that are currently owned by that node
- Default mode is to only run when an error is detected on any of the monitored devices
- Can also be run at a fixed time interval. This is controlled by the '-t' option
- Because of its asynchronous nature mdmonitord has exposed many locking issues.
  - Turning on '-t' option at a fairly short interval while running the Tslvm test suites is a good way to test locking changes.

rpc.metad

- A daemon functioning as a server process that is used to manage local copies of metadevice diskset information.
- Updates the following diskset information:
  - Creates/deletes diskset entries
  - Creates/deletes node name entries
  - Creates/deletes device name entries
- Commands contact rpc.metad for any/all diskset information.
- Only rpc.metad reads the USER records in the local metadb

Mediator Review

- A method of achieving quorum and allowing a diskset to be taken when only half of the diskset replicas are available.
- Originally implemented to support HA configurations of 2 hosts and 2 strings of disks.
- A mediator host is a host with SVM installed that is running rpc.metamedd(1M) and has been added to a diskset with the 'metaset -a -m' command. The mediator host participates in checking the mediator quorum.
  - Any host can be a mediator. It is not necessary that the mediator host be a member of the diskset; only that it runs rpc.metamedd.

rpc.metamedd

- A daemon functioning as a server process that is used to manage mediator information.
- Creates/deletes diskset mediators via the metaset command. This information is stored in the local replica.
- Queries and updates the state of the mediators through kernel rpc calls.
  - Called to get mediator host information when it is necessary to use mediators to establish quorum.
  - Updates the state of the mediators when the diskset replicas are updated.

Establishing Mediator Quorum

- The function to establish mediator quorum is 'mediate' in usr/src/uts/io/lvm/md/md_mddb.c and is called when a request is made to take a diskset and the number of available replicas is 50%.

Mediate Code Flow

- If there are no mediators for the diskset then the set is stale (read-only).
- Contact each of the mediator hosts for the mediator data. If none respond then the set is stale (read-only).
- Find the maximum commit count in the mediators that responded. All of the mediators that do not have this commit count are removed from consideration.
- If either of the following is true then mediator quorum will be achieved and the set take will succeed:
  - Any of the remaining mediators are marked as golden.
  - The number of remaining mediators is ½ of the total + 1.

Updating Mediator Information

- Mediator Information is updated when the locator block in the replica is updated. This is done when:
  - A replica is created
  - A replica is deleted
  - An error is detected on a replica
- The function to update the mediator information is 'upd_med' in usr/src/uts/io/lvm/md/md_mddb.c.

upd_med Code Flow

- If there are no mediators then do nothing.
- If this is a Multi-Owner diskset and this node is not the master then do nothing. Mediator updates are only done on the master node.
- Update the commit count on each of the mediator hosts.
- If the diskset does not have replica quorum and does not have mediator quorum then panic.
- If mediator quorum is achieved and exactly ½ of the replicas are available then mark the mediators as golden.

rpc.metamhd

- Takes the SCSI reservations on the disks in a set when drives are added or the set is taken
- Releases the SCSI reservations when drives are deleted from a set or when the set is released
- rpc.metamhd is not used but still run when operating in Oban/SunCluster environment. SunCluster takes care of SCSI reservations.
- Legacy code. No significant changes since SDS 4.2.
- No inter-node communication. Not sure why this is an rpc daemon ...

SVM Daemon Debug

- Connect mdb to a running process
  - mdb -p `pgrep <daemon name>`
    - Stops the daemon and allows the setting of a breakpoint
- pstack on hung daemons
- Command line options
  - 'mdmonitord -d <debug level>'
    - Debug levels 0 – 9 (higher number, more output)
    - Debug level of 9 will not run mdmonitord in background

SVM Services And Daemons