



SMF

Service Management Facility

Codename: Greenline

Jarod Nash
Systems TSC Kernel
Sun Microsystems



Agenda

- SMF Background, Advantages and Architecture
- Commands Summary
- Milestones, Contracts, Booting and Profiles
- Writing an SMF service manifest
- Live Demo
- SMF ARC Policy
- Other Stuff

SMF: A Child of FMA

- Before the FMA project began, it was recognised that Solaris needed an automated response to H/W faults
 - > eg UE (Uncorrectable Error) detected by hardware
 - > Pre-Solaris 10: Identify whether UE impacts kernel or userland
 - kernel – panic
 - userland – kill process and reboot
 - > Reboot as we have no knowledge of what was killed
 - > No knowledge of interdependencies
 - > Services are not monitored
 - > Services are stateless
- SMF defines, monitors and restarts system services to provide an automated response (*Self Heal*)

SMF: Advantages

- In addition to this H/W resilience, SMF also offers:
 - > Recovery from SysAdmin mistakes, ie killing wrong daemon
 - > Clear dependencies
 - > Services start when dependencies met
 - > Dependents can be set to restart if required
 - > Services start in parallel
 - > Faster boot times (65% NQF)*
 - > Central configuration database: *Repository*
 - > System is quieter when booting
 - > Services write to their own log files

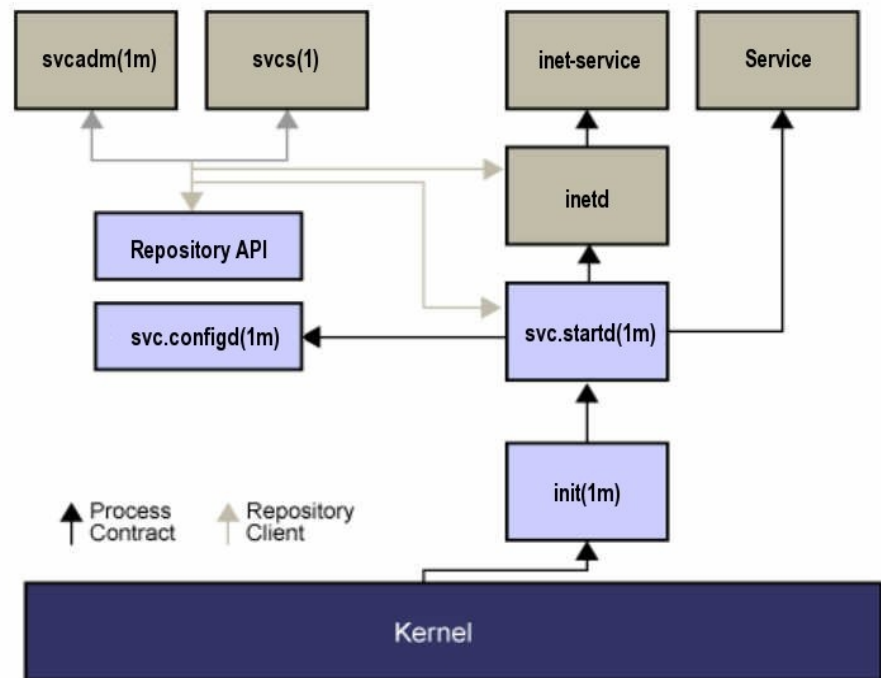
*Non-Qualified Figure

Based on Marketing information at time of launch.

Your mileage will vary depending upon hardware configuration and service workload

SMF: Architecture

- New daemons
 - > svc.startd
 - > svc.configd
- inetd integrated into SMF
- New commands:
 - > svcs(1), svcadm(1m),
 - > svccfg(1m), svcprop(1)
- Centralised log files
 - > /var/svc/log/FMRI.log
- FMRI names for services



SMF: Terminolgy

- FMRI (Fault Managed Resource Identifier)
 - > Name of the service, eg `svc:/system/system-log:default`
- Service Instance
 - > Running version of a service. Most instances are *default*
- Restarter
 - > Service responsible for restarting a service: `svc.startd/inetd`
- Dependency
 - > Formal description of the other services that are required to start a service
- Contract
 - > New process notification mechanism used by restarters

SMF: Terminolgy

- Manifest
 - > Description and initial configuration file for a service or set of related services. Delivered with the product
 - > Written in XML
- Repository
 - > Configuration database for all services. Allows for settings to remain persistent across reboot
- Milestone
 - > A way to group services together. If services are like files, then milestones are like directories

SMF: Key Commands

svcs

- Display service(s) state
- Supports pattern matching, eg `svcs '*print*'`
- Useful usage:
 - `svcs` - show state of all enabled services
 - `svcs -xv` - used to debug non-running services
 - `svcs -d <FMRI>` - show service dependencies
- Examples:
 - `# svcs -xv nfs/server`
 - `# svcs '*print*'`

SMF: Key Commands:

svcadm

- Control services and milestones
- Useful usage:
 - enable/disable - permanently, or use "-t" for temporarily
 - refresh - refresh config, run optional refresh method
 - restart - run stop, then start methods
 - milestone - move system to specified milestone
 - clear - clear *maintenance* flag and retry
- Use *clear* to retry a service when failed
- Examples:
 - # `svcadm refresh system-log`
 - # `svcadm disable -t name-service-cache`

SMF: Key Commands

svccfg

- Used to access SMF Repository
- Import new service definitions from XML files
- Modify definition in the SMF Repository
 - > Does not modify XML definition
- Remove service definitions
- Changes are persistent across reboots

SMF: Key Commands

svccprop

- Retrieve properties from SMF Repository
- Useful for method scripts to extract service properties
 - > Avoids nasty “hacks” which can be lost via patching
 - > For example, rather than edit /etc/rc2.d/S80lp, we now use properties and svccprop in start method:

```
fd_limit=`/bin/svccprop -p lpsched/fd_limit ${svc}
```

- Also useful in debugging:

```
# svccprop -p start FMRI_of_failing_svc
...
start/exec astring /lib/svc/method/svc-start-script
...
```

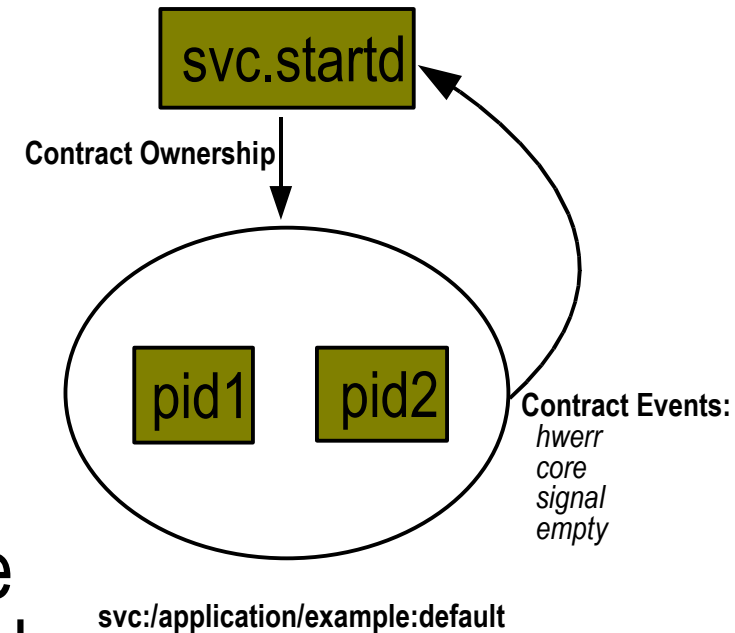
SMF: Not Quite Key Commands

inetconv/inetadm

- inetd now a *delegated restarter*
- Configuration stored in Repository, not inetd.conf
- **inetconv** provided to ease transistion to Repository
 - > Generates XML file for inetd.conf entries
- **inetadm** is an inetd specific admin tool
 - > Combined svcs/svccfg roles
 - > inetadm -d telnet – disable svc, useful for recent Bug: 6523815
 - > inetadm -l telnet – list svc properites
 - > inetadm | grep telnet – report svc state
 - > When new to SMF, easier to stick to generic SMF commands

SMF: Contracts

- New functionality in Solaris 10 which notifies `svc.startd` when something *bad* happens to a process which is part of a service
- Works by grouping processes together and generating events, rather than polling to check state
- `svc.startd` can then restart service by running `stop`, then `start` method
- Too many failures will result in the service being put into the *maintenance* state



SMF: Milestones

Milestone

none

single-user

multi-user

multi-user-server

all

Solaris Admin

boot -s



^D to finish

SMF Admin

boot -m milestone=*none*



svcadm milestone *all*

- 3 milestones relate to existing run levels
- 2 pseudo milestones
- Use existing tools for Solaris administration
- Only use the *none* and *all* milestones for SMF administration
- Start methods for su/mu/mus run /sbin/rc[S,2,3] scripts

SMF: More Notes on Booting

- SMF “-m” boot flag understands:
 - > milestone – see previous slide
 - > verbose
 - > single line output for each service state change
 - > debug
 - > An impossible amount of information, detailing at a function by function level the activity within svc.startd
- Almost always not what you want (*old school*)
 - > Debug the service, not the boot:
 - > svcs -xv, service log file, svcprop -p start, svcadm clear

Profiles

- Description of the services that are to be used on a system
 - > Processed in order: generic, platform, site
 - > Profiles may include sub-profiles, eg: ns, inetd
- Each profile is applied once
 - > Can apply profile at any time with:

```
# cd /var/svc/profile
# svccfg apply ns_none.xml
# svccfg apply ns_nis.xml
```
- Never modify existing profiles
 - > site.xml is for local customisations/JumpStart

Writing an SMF Manifest

- Start with “Service Developer Introduction”
 - > BigAdmin link from OpenSolaris SMF Community
- Also check example SMF Community manifests:
 - > <http://opensolaris.org/os/community/smf/manifests/>
- Outlines 12 steps:
 - > 1. Name your service, 2. Identify whether your service may have multiple instances, 3. Identify your service model, 4. Identify how your service is started/stopped, 5. Determine faults to be ignored, 6. Identify dependencies, 7. Identify dependents, 8. Insert your service into a milestone, 9. Create, if appropriate, a default instance, 10. Create template information to describe your service, 11. Write/update an administrative command, 12. Remove your script from /etc/rc?.d locations and /etc/init.
- Or, take a copy of existing manifest and *edit...*

SMF Manifest for *littled*

- **littled** is a simple daemon
 - > Configuration file: `/var/tmp/littled.conf`
 - > Listens on port 13567, with commands:
 - > `prtconfig`, `status`, `bye`, `die`, `signal`, `core`, `udue` (where available)
- Copy **utmp.xml**, edit and change:
 - > Manifest Name: **SUNWcsr:utmpd** -> **JN:littled**
 - > Name: **system/utmp** -> **application/littled**
 - > Dependent Name: **utmpd_...** -> **littled_...**
 - > Exec method: **/lib/svc/method/svc-utmpd** -> **/bin/littled**
 - > Template: **utmpx monitoring** -> **littled daemon**
 - > Documentation: Delete **utmpd(1m)/utmpd(4)** references

SMF Demo...

SMF ARC Policy

- Work done in the last 6-8 months
- Policy requires no new files or modifications in /etc directories and files (exception: *private* config files)
- Legacy (/etc/rc?.d/*) services must switch to SMF before changes are approved
- Guidelines:
 - Disabled by default, use least privilege, provide RBAC authorizations, use profiles where appropriate, template info, distinct/structured Repository property naming
- ARC consultation required for *complex* configuration

Other Stuff

- Service types: Legacy, Contract, Transient, Wait
- Service relationships: restarting dependants
- Method variables and tokens
- Delegated administration with RBAC
- OpenSolaris SMF Community
 - > Overview, FAQ, Developer Guide, Upcoming Work
 - > smf-discuss@opensolaris.org alias