Solaris 10 Workshop
Service Management Facility

Bob Netherton

Solaris Adoption, US Client Solutions

Sun Microsystems

Agenda

- Motivation for SMF
- Core concepts and terminology
- Administrative Commands
- Migrating a Legacy RC service
- New Boot Process
- Troubleshooting/Recovery
- Resources

Learning Goals

By the end of this module you should ..

- Have an understanding of the core concepts and commands that comprise SMF
- Use commands to enable, disable, restart or recover a service.
- Boot a system using SMF
- Manage a legacy service
- Migrate a service to SMF
- Restore a corrupted SMF database

SMF First Impression

- Solaris has a wealth of innovations
  - Most of these features are optional, valuable – but optional
- You cannot avoid SMF
  - You experience it on first boot (loading service definitions)
  - /etc/init.d is significantly smaller than in Solaris 9
  - Where did all of those service scripts go ?
- Legacy methods still work
  - Sequencing is very interesting as you migrate services
- Argh!!! - Why did you do all this to me ?

SMF First Impression

- SMF seems more complicated than it really is
- Migrate a legacy RC service to be immersed
  - The first one seems hard
  - The second one is a snap
- SMF rocks!

What is a Service ?

Definition:

A long lived software object with a well-defined state, error boundary, definition of start and stop, and relationship to other services. A service is often critical to operation of system or fulfillment of business objectives.

- A collection of processes (aka daemons)
- A set of configuration files
- Management utilities

How are services started today ?

init(1M) via rc scripts in /etc/init.d

- Long time running or one time initializations

inetd(1M) as defined by inetd.conf(4)

- Short-lived to provide transient network functions

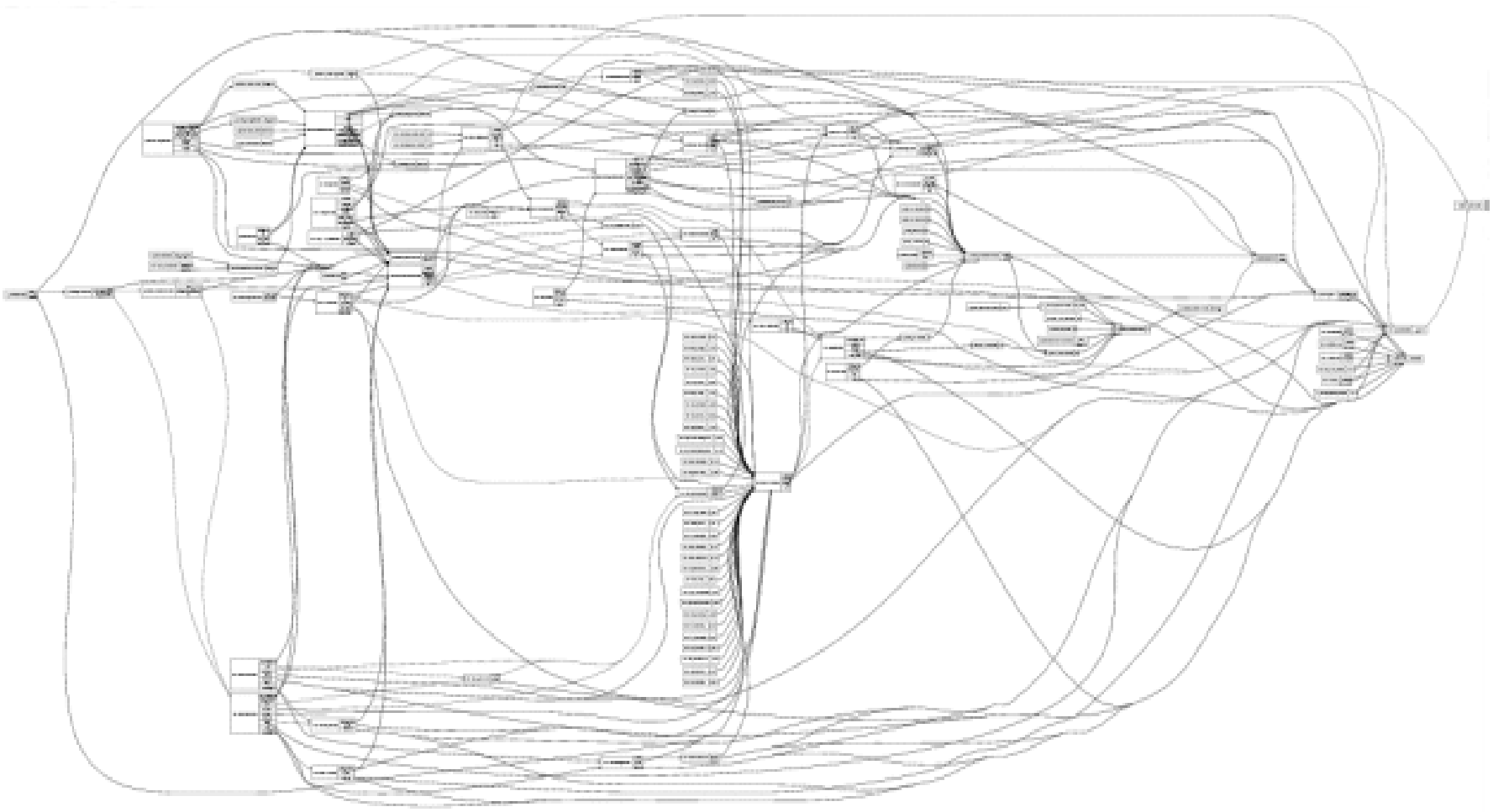
init(1M) as defined by /etc/inittab

- Restartable or one time functions

What about ...

- Dependencies
- Detailed status
- Services or dependencies that may span multiple hosts (a grid, et al)
- Restarting services (correctly)
- Multiple instances of a service

Solaris Dependencies today



Service Management Today

- Multiple administration techniques
- Inconsistent dependencies, often unknown
- Lack of well defined error boundaries
- Service Oriented Architectures (SOA) require a more robust definition of services
- A new system is required

Service Management in Solaris 10

- A service is now a first class object that can be managed and observed
- All services have a common framework
- Legacy mechanisms still work
- Automated restart of services in correct order:
  - administrative error
  - software bug
  - uncorrectable hardware error
- Parallel startup improves system boot time

Service Management in Solaris 10

- Easy access to information about misconfigured/misbehaving services
  - Easy to script!
- Admins can get a meaningful system view
- Changes persist across upgrades and patches
- Securely delegate tasks to non-root users
- Snapshots and repository backup taken automatically: restore, undo

Predictive Self-Healing

Solaris Fault Manager (FMA)

- Solaris Fault Manager provides detection and diagnosis of errors, leading to isolation or deactivation of faulty components and precise, accurate administrative messaging.

Solaris Service Manager (SMF)

- smf(5) makes Solaris services self-healing.

Hardware faults which previously caused system restart are now isolated to the affected services. Services are also automatically restarted in the face of hardware and software faults.

SMF Core Concepts

SMF service definition

SMF identifiers

Service States

SMF Manifests

Service Configuration Repository

SMF snapshots

SMF compatibility

SMF Service

- Definition

  One or more daemons and/or configurations previously configured, started or stopped through System V RC scripts, inetd or /etc/inittab

- A service may have multiple instances
  - A web service
  - A web server (httpd)
  - IPv4 and IPv6

Service Names in Solaris 10

FMRI – Fault Management Resource ID

svc://localhost/network/login:rlogin

SMF Identifiers
naming and syntax

Scheme:

svc – SMF managed service

lrc – legacy RC script

SMF Identifiers
naming and syntax

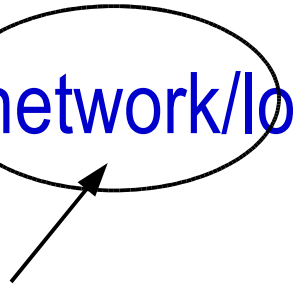
Location:

localhost

<hostname> in future release

SMF Identifiers
naming and syntax

svc://localhost/network/login:rlogin



Functional category:

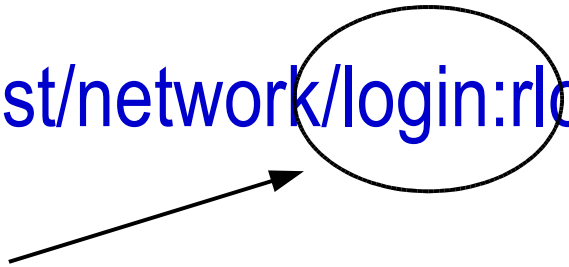
- application
- system
- device
- network
- milestone
- platform
- site

SMF Functional Categories

- Application – traditional daemon
- Device – useful for dependencies
- Milestone – similar to SVR4 run levels
- Network – inetd converted services
- Platform – platform specific services
- System – platform independent system services
- Site – reserved for a future use

SMF Identifiers
naming and syntax

svc://localhost/network/login:rlogin

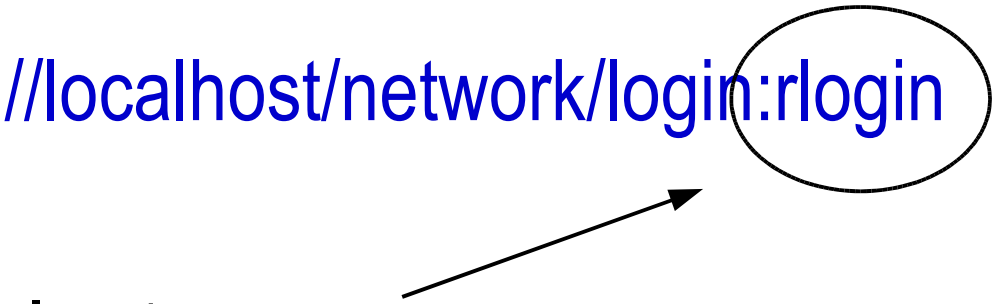


Description:

Relate to method or RC script

SMF Identifiers
naming and syntax

svc://localhost/network/login:rlogin



Instance:

default is the ..... default instance

SMF Identifiers
aliases and examples

svc://localhost/network/login:rlogin

svc:/network/login:rlogin

network/login:rlogin

rlogin

svc://localhost/system/system-log:default

svc:/system/system-log:default

svc:/platform/i86pc/kdmconfig:default

Service States

online – The service instance is enabled and has successfully started.

offline – The service instance is enabled, but the service is not yet running or available to run.

disabled – The service instance is not enabled and is not running.

maintenance – The service instance has encountered an error that must be resolved by the administrator.

Service States

legacy_run – The legacy service is not managed by SMF, but the service can be observed. This state is only used by legacy services.

degraded – The service instance is enabled, but is running at a limited capacity.

uninitialized – This state is the initial state for all services before their configuration has been read.

Service Dependencies

- Dependencies can be to another SMF managed service or a file
- Types of dependencies
  - require_all – all must be online or degraded
  - require_any – at least one online or degraded
  - optional_all – all are online, disabled, degraded, or in maintenance
  - exclude_all – all are disabled, in maintenance, or not present (files)

Dependency actions

- What do to when a dependent service changes state
  - Service failure
  - Administrative stop
  - Refreshed due to property changes
- Controlled by restart_on attribute of dependency

	restart_on attribute			
Reason for Service Stop	None	Error	Restart	Refresh
Error	No	Yes	Yes	Yes
Non error stop	No	No	Yes	Yes
Refresh	No	No	No	Yes

Dependency Examples

```
<dependency name='network'    grouping='require_any'    restart_on='error'    type='service'>
        <service_fmri value='svc:/milestone/network' />
</dependency>

<dependency name='nlockmgr'  grouping='require_all'   restart_on='error'   type='service'>
        <service_fmri value='svc:/network/nfs/nlockmgr' />
</dependency>

<dependency name='mapid'    grouping='optional_all'    restart_on='error'    type='service'>
       <service_fmri value='svc:/network/nfs/mapid' />
</dependency>

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

<dependency name='keyserv'    grouping='optional_all'    restart_on='none'    type='service'>
        <service_fmri value='svc:/network/rpc/keyserv' />
</dependency>
```

Basic Commands

svcs(1) – detailed state information about all service instances

svcadm(1M) – common service management tasks (enable, disable, ...)

svccfg(1M) – manipulate the SMF repository

svcprop(1) – view SMF repository data

inetadm(1M) – view and configure inetd managed services

inetconv(1M) – convert an inetd.conf(4) to an SMF manifest

Service States

```
#svcs -a  (edited for brevity)
 STATE          STIME    FMRI
 legacy_run     Oct_26   lrc:/etc/rcS_d/S50sk98sol
 legacy_run     Oct_26   lrc:/etc/rc2_d/S10lur
 legacy_run     Oct_26   lrc:/etc/rc3_d/S90samba
 disabled       Oct_26   svc:/system/metainit:default
 disabled       Oct_26   svc:/network/rpc/nisplus:default
 online         Oct_26   svc:/system/svc/restarter:default
 online         Oct_26   svc:/milestone/name-services:default
 online         Oct_26   svc:/platform/i86pc/eeprom:default
 online         Oct_26   svc:/system/filesystem/minimal:default
 online         Oct_26   svc:/network/physical:default
 online         Oct_26   svc:/milestone/single-user:default
 online         Oct_26   svc:/network/rpc-100083_1/rpc_tcp:tcp
 online         Oct_26   svc:/network/rpc/rstat:udp
 offline        Oct_26   svc:/application/print/ipp-listener:default
 offline        Oct_26   svc:/application/print/rfc1179:default
 maintenance    Oct_26   svc:/network/rpc/keyserv:default
```

View detailed status of a service

% svcs -l <fmri>

% **svcs -l ssh**

```
fmri         svc:/network/ssh:default
name         Secure Shell
enabled      true
state        online
next_state   none
restarter    svc:/system/svc/restarter:default
contract_id  24
dependency   require_all/restart
file://localhost/etc/ssh/sshd_config (-)
dependency   require_all/none svc:/system/cryptosvc (online)
dependency   require_all/none svc:/network/loopback (online)
dependency   require_all/none svc:/system/filesystem/usr:default
(online)
```

SMF common view of an inetd managed service

% svcs -l network/smtp:sendmail

```
fmri            svc:/network/smtp:sendmail
enabled         true
state           online
next_state      none
restarter       svc:/system/svc/restarter:default contract_id 29462
dependency      require_all/refresh file://localhost/etc/nsswitch.conf (-)
dependency      require_all/refresh file://localhost/etc/mail/sendmail.cf (-)
dependency      optional_all/none svc:/system/system-log (online)
dependency      require_all/refresh svc:/system/identity:domain (online)
dependency      require_all/refresh svc:/milestone/name-services (online)
dependency      require_all/none svc:/network/service (online)
dependency      require_all/none svc:/system/filesystem/local (online)
```

Who is dependent on me ?

% svcs -D network/physical

```
STATE          STIME    FMRI
disabled       20:18:22 svc:/network/dns/client:default
disabled       20:18:22 svc:/network/dns/server:bind9
disabled       20:18:22 svc:/network/dns/server:default
disabled       20:18:24 svc:/network/rpc/bootparams:default
disabled       20:18:24 svc:/network/nfs/server:default
disabled       20:21:01 svc:/network/shell:kshell
online         20:18:33 svc:/application/print/cleanup:default
online         20:20:31 svc:/system/identity:domain
online         20:20:31 svc:/system/identity:node
online         20:20:32 svc:/network/initial:default
online         20:20:33 svc:/network/nfs/status:default
online         20:20:33 svc:/network/nfs/mapid:default
online         20:20:33 svc:/milestone/single-user:default
online         20:20:33 svc:/network/nfs/nlockmgr:default
online         20:20:58 svc:/network/inetd:default
online         20:21:02 svc:/network/shell:tcp
online         20:21:02 svc:/network/shell:tcp6only
online         20:21:02 svc:/network/nfs/client:default
```

What are my dependencies ?

% svcs -d network/inetd

```
STATE          STIME    FMRI
disabled       20:18:22 svc:/network/inetd-upgrade:default
online         20:18:23 svc:/milestone/name-services:default
online         20:18:24 svc:/network/loopback:default
online         20:20:30 svc:/network/physical:default
online         20:20:32 svc:/network/rpc/bind:default
online         20:20:33 svc:/milestone/single-user:default
online         20:20:57 svc:/system/filesystem/local:default
```

% svcs -l network/inetd | grep ^depend

```
dependency   require_any/error svc:/network/loopback (online)
dependency   require_all/error svc:/system/filesystem/local (online)
dependency   optional_all/error svc:/milestone/network (online)
dependency   optional_all/error svc:/network/rpc/bind (online)
dependency   optional_all/none svc:/network/inetd-upgrade (disabled)
dependency   require_all/none svc:/milestone/sysconfig (online)
   svc:/milestone/name-services (online)
```

Lab #1 – examine a service

1) Let's look at my favorite service: gdm2-login
2) Is it running ?
3) What are it's dependencies
4) What services are dependent on it ?
5) Who is it's designated restarter ?

Lab #1 – examine a service

- Is it running ?

  # svcs gdm2-login

```
STATE          STIME    FMRI
online         22:33:28 svc:/application/gdm2-login:default
```

- What are it's dependencies

  # svcs -d gdm2-login

```
STATE          STIME    FMRI
online          8:02:43 svc:/system/filesystem/local:default
online          8:02:48 svc:/system/utmp:default
```

Lab #1 – examine a service

- What services are dependent on it ?

  # **svcs -D gdm2-login**

- Who is it's designated restarter ?

  # **svcs -l gdm2-login | grep ^restarter**

Enable a service

- Become superuser or assume a role that includes the Service Management Profile.

  # svcadm enable network/login:rlogin

  # svcs -l network/login:rlogin

```
fmri            svc:/network/login:rlogin
name            The remote login service.
enabled         true
state           online
next_state      none
restarter       svc:/network/inetd:default
```

- Use **svcadm enable -r** to enable a service plus all of its dependents

Temporarily Disable a Service

- Become superuser or assume a role that includes the Service Management Profile.
- Check the dependents of the service you want to disable

  # svcs -D [fmri]

  # svcs -D network/login:rlogin

  # svcadm disable -t network/login:rlogin

  # svcs network/login:rlogin

```
STATE    STIME  FMRI
disabled   11:17:24  svc:/network/login:rlogin
```

- Not persistent across reboots

Permanently Disable a service

- Become superuser or assume a role that includes the Service Management Profile.
- Check the dependents of the service you want to disable

  # svcs -D [fmri]

  # svcs -D network/login:rlogin

  # svcadm disable network/login:rlogin

  # svcs network/login:rlogin

```
STATE    STIME  FMRI
disabled    11:17:24  svc:/network/login:rlogin
```

- Enabled dependent services will go offline

Restart a service

- Become superuser or assume a role that includes the Service Management Profile.

  # **svcadm restart FMRI**

- The service method is responsible for any synchronization between shutdown and startup

Restoring from maintenance

- Become superuser or assume a role that includes the Service Management Profile.
- Determine if any processes from the service are still running.

  # **svcs -p fmri**

  will output all pid's associated with the service
- Terminate any leftover processes or daemons

  # **kill** pid(s)
- Fix the actual service problem
- Return the service to the default state

  # **svcadm clear fmri**

Lab #2 – GDM

- Turn off dtlogin

  # **/usr/dt/bin/dtconfig -d**

  # **/usr/dt/bin/dtconfig -kill**
- Log in as root
- Enable gdm

  # **svcadm enable -t gdm2-login**
- Log in a few times and validate
- Reboot and describe what happens

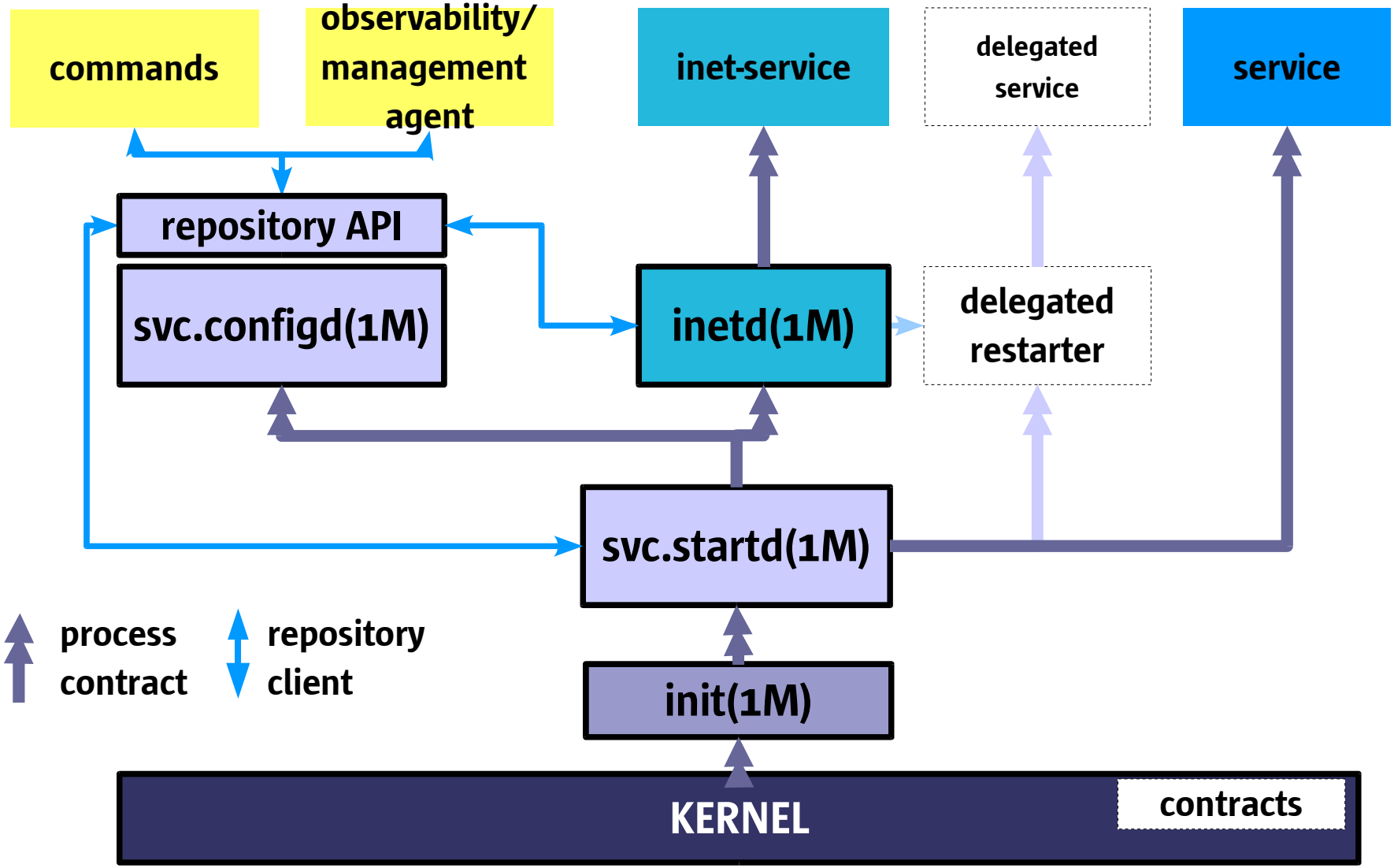
Lab #2 – Using gdm

1) What happened ?
2) What graphical login greeter is running ?
3) Why ?
4) How do you restore your graphical environment ?

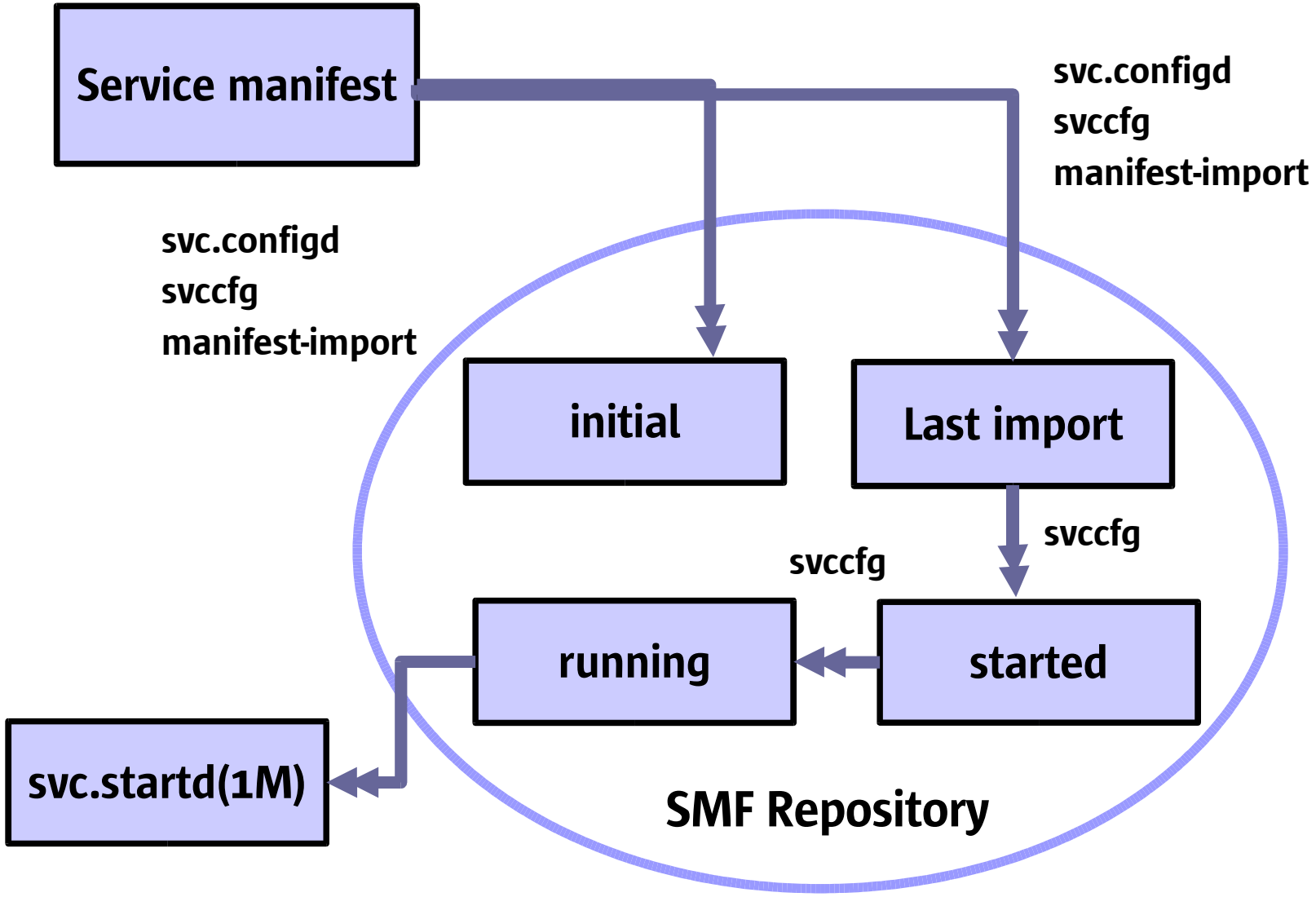
Lab #2 – Using gdm

- What happened ?
  - gdm replaces dtlogin as the login manager
  - No graphical login after reboot
- Why ?
  - The gdm2-login service was enabled temporarily
- How do you restore a graphical environment ?
  - dtconfig -e or svcadm enable gdm2-login

Components: Architecture schematic



Components: Manifest and Repository



SMF Master Restarter Daemon

- /lib/svc/bin/svc.startd
- Reads the SMF repository
- The master process starter and restarter
  - Restarts services that have failed
  - Shuts down services whose dependencies are no longer satisfied
  - Runs legacy rc scripts at run level transitions
- Provides system view of service status

SMF Manifests

- Located in /var/svc/manifest
- Complete XML description of a service
- Loaded into the repository at boot time
- Use svccfg(1M) to manually import a service definition (aka service bundle)
- Manifest DTD at

  /usr/share/lib/xml/dtd/service_bundle.dtd.1
- See service_bundle(4) man page
- ISVs should supply a service bundle

SMF Manifests

```
<service_bundle type='manifest' name='SUNWzoner:zones'>

<service  name='system/zones'  type='service' version='1'>

<create_default_instance enabled='false' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<exec_method type='method' name='start' exec='/lib/svc/method/svc-zones %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/lib/svc/method/svc-zones %m' timeout_seconds='500'> </exec_method>

<property_group name='startd' type='framework'>
            <propval name='duration' type='astring' value='transient' />
</property_group>

<stability value='Unstable' />

<template>
    <common_name>
        <loctext xml:lang='C'>  Solaris zones </loctext>
    </common_name>

    <documentation>
        <manpage title='zones' section='5' manpath='/usr/share/man' />
        <manpage title='zoneadm' section='1M' manpath='/usr/share/man' />
    </documentation>
</template>
</service>
</service_bundle>
```

Name of the service

Default state

Dependency

Start and Stop methods

SMF Configuration Repository

- Located in /etc/svc
- Local zones have their own repository
- Persistent configuration information as well as runtime properties for services
- Distributed among local memory (volatile) and local files (repository.db).
- May be placed in a directory (LDAP) in a future Solaris release

SMF Repository Administration

- svccfg(1M) to modify the repository
- svcprop(1M) to view the repository
- libscf(3LIB) provides repository APIs
- svc.configd(1M) is the repository daemon
  - > Run at boot time to adjust properties
  - > Restarted upon any SMF failure

SMF Repository Profile

- Set of service instances and enable property
- Generated by svccfg extract
- Activated by svccfg apply
- System profiles located in

  /var/svc/profile
- Useful for copying service states between systems

SMF Repository Profile

```
# svccfg extract
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='profile' name='extract'>
  <service name='system/console-login' type='service' version='0'>
    <instance name='default' enabled='true'/>
  </service>
  <service name='system/device/local' type='service' version='0'>
    <instance name='default' enabled='true'/>
  </service>
  <service name='milestone/devices' type='service' version='0'>
    <instance name='default' enabled='true'/>
  </service>
  <service name='system/identity' type='service' version='0'>
    <instance name='domain' enabled='true'/>
    <instance name='node' enabled='true'/>
  </service>
  <service name='system/filesystem/local' type='service' version='0'>
    <instance name='default' enabled='true'/>
  </service>
```

SMF Repository Archive

- Complete set of persistent data for all service instances
  - > Dump in XML format similar to manifest
  - > Does not include transient properties
- Generated by svccfg archive
- Useful for copying service definitions between systems

SMF Repository Archive

```
# svccfg archive
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM
   '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='archive' name='none'>
  <service name='system/console-login' type='service'
   version='0'>
    <create_default_instance enabled='true'/>
    <single_instance/>
    <dependency name='fs' grouping='require_all'
   restart_on='none' type='service'>
      <service_fmri
   value='svc:/system/filesystem/minimal'/>
    </dependency>
    <dependency name='identity' grouping='require_all'
   restart_on='none' type='service'>
      <service_fmri value='svc:/system/identity:node'/>
    </dependency>
    <dependency name='utmpx' grouping='require_all'
   restart_on='none' type='service'>
      <service_fmri value='svc:/system/utmp:default'/>
    </dependency>
```

SMF snapshots

- Complete collection of properties for a service
- Historical view of a service instance
- Simplifies rollback of service configuration changes
- Created automatically or manually as needed

SMF snapshots

- Standard snapshots
  - *initial* – taken on the first import of the manifest
  - *last_import* – taken during last import
  - *running* – properties of the currently running service instance
  - *start* – taken at the last successful start
- Can create manually with svccfg(1M)

SMF snapshots

- Use svccfg(1M) to view snapshot properties
- To activate a snapshot
  - > svccfg(1M) to select snapshot
  - > svcadm refresh to update svc.startd
  - > svcadm restart to start service with new property values

Revert to a previous snapshot

```
# svccfg
   svc:>
   svc:> select system/console-login:default
   svc:/system/console-login:default>
   svc:/system/console-login:default> listsnap
   initial
   running
   start
   svc:/system/console-login:default>
   svc:/system/console-login:default> revert start
   svc:/system/console-login:default>
   svc:/system/console-login:default> quit
# svcadm refresh system/console-login
# svcadm restart system/console-login
```

Modify a service property

- Make changes to the config file
  - > Could be updates to repository (svccfg)
  - > If modifying a snapshot, refresh the service
- Restart the service

  # svcadm restart FMRI

Legacy RC scripts

- Identified by the lrc scheme
  Example: lrc:/etc/rcS_d/S35cacheos_sh
- Limited SMF management
- Start status only
- Executed on run level transitions after SMF managed services
- Allows third party applications to run without modification

Concepts Review

- Each service on the system has a unique FMRI
- XML manifests are kept for each service
- Snapshots of configuration data are kept in the repository
- Legacy /etc/rc*.d scripts are still run

Perform MySQL setup

- See /etc/sfw/mysql/README.solaris.mysql
  - There is a typo in the installation instructions
  - Line 15 should read
    - **chmod -R 770 /var/mysql**
- Start database and test using mysqladmin

```
# /etc/sfw/mysql/mysql.server start
Starting mysqld daemon with databases from /var/mysql

# mysqladmin status
Uptime: 32  Threads: 1  Questions: 1  Slow queries: 0  Opens: 6
Flush tables: 1  Open tables: 0  Queries per second avg: 0.031
```

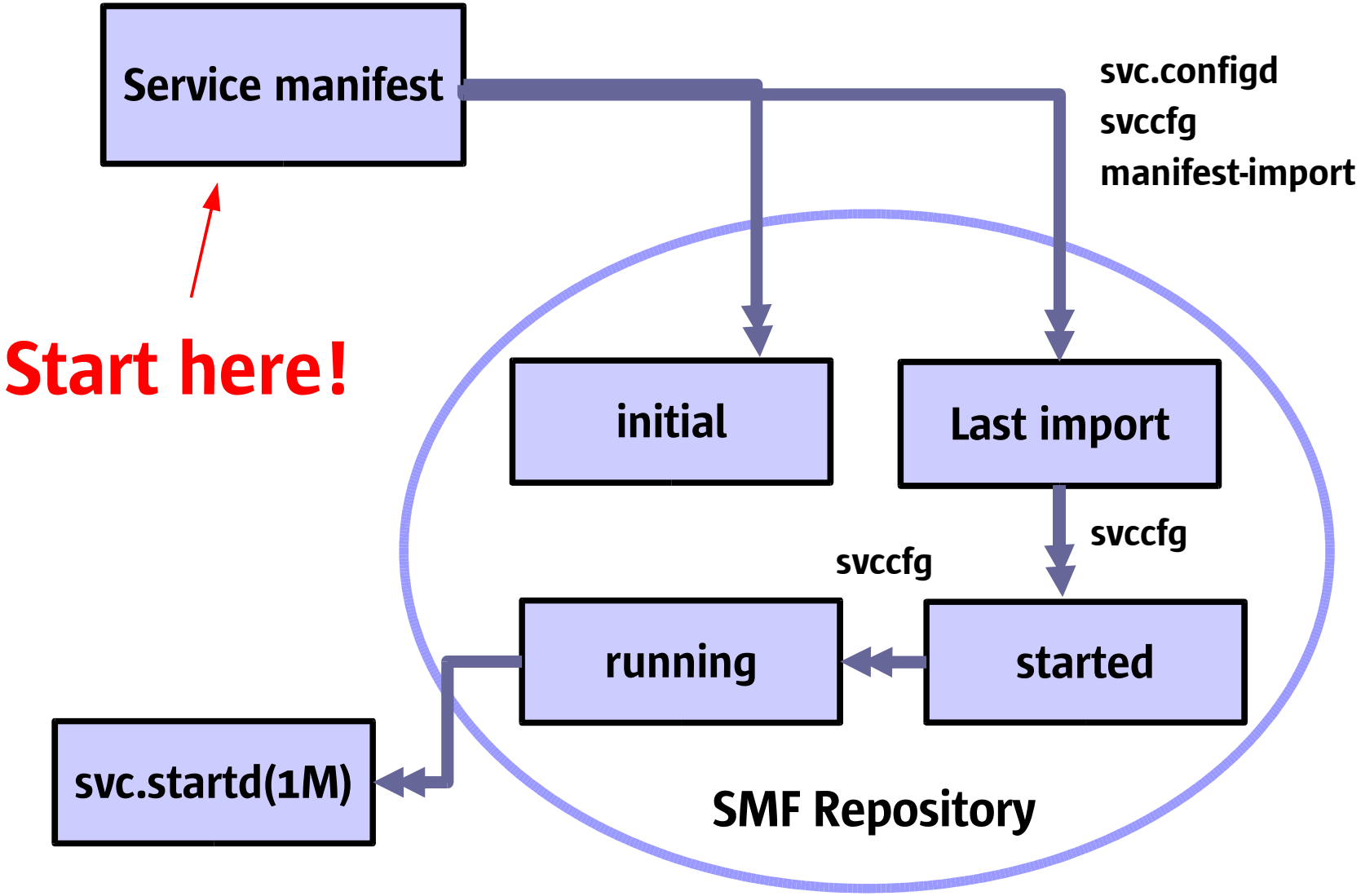
Perform MySQL setup

- Stop the database!
  - > We don't want SMF to try to start it if it is already running

```
# /etc/sfw/mysql/mysql.server stop
/etc/sfw/mysql/mysql.server stop Killing mysqld with pid 1212
Wait for mysqld to exit.050907 21:16:19  mysqld ended

 done
```

- Don't link the RC script in /etc/rc3.d

Components: Manifest and Repository



Creating an SMF Manifest

- XML file describing service properties
- Located in /var/svc/manifest/<dir>
  - site is recommended for locally developed services
- DTD at /usr/share/lib/xml/dtd/service_bundle.dtd.1
- service_bundle(4) man page

Why reinvent the wheel?

Copy and modify an existing service manifest!

Some suggested sources of manifests

- Explore /var/svc/manifest for similar services
  - system/utmp is a simple standalone daemon
  - system/coreadm is a simple configuration service, and
  - network/telnet is an inetd-managed daemon
- Initial inet service manifests can be created easily by invoking: `inetconv -i <file>`
- DTD is self-documenting; read it at
  `/usr/share/lib/xml/dtd/service_bundle.dtd.1`

Example: Zones Manifest

```
<service_bundle type='manifest' name='SUNWzoner:zones'>

<service  name='system/zones'  type='service' version='1'>

<create_default_instance enabled='false' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<exec_method type='method' name='start' exec='/lib/svc/method/svc-zones %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/lib/svc/method/svc-zones %m' timeout_seconds='500'> </exec_method>

<property_group name='startd' type='framework'>
            <propval name='duration' type='astring' value='transient' />
</property_group>

<stability value='Unstable' />

<template>
    <common_name>
        <loctext xml:lang='C'>  Solaris zones </loctext>
    </common_name>

    <documentation>
        <manpage title='zones' section='5' manpath='/usr/share/man' />
        <manpage title='zoneadm' section='1M' manpath='/usr/share/man' />
    </documentation>
</template>
</service>
</service_bundle>
```

Name of the service

Default state

Dependency

Start and Stop methods

Documentation

Step 1: Start with an existing manifest

- Copy a manifest that most closely resembles your new service
  - Put in /var/svc/manifest/local
- Change the following sections
  - <service name=”your service name”>
  - <create_default_instance enabled=true | false>
  - <template> to provide text descriptions of service name and man page locations.

Our new manifest – so far

```
<service_bundle type='manifest' name='MySQL'>

<service  name="application/mysql'  type='service' version='1'>

<create_default_instance enabled='true' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<exec_method type='method' name='start' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
<exec_method type='method' name='restart' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>

<property_group name='startd' type='framework'>
            <propval name='duration' type='astring' value='transient' />
</property_group>

<stability value='Unstable' />

<template>
    <common_name>
        <loctext xml:lang='C'>  MySQL Database </loctext>
    </common_name>

    <documentation>
        <manpage title='mysql' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqld' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqladmin' section='1' manpath='/usr/sfw/man' />
    </documentation>
</template>
</service>
</service_bundle>
```

Name of the service

Default state

Documentation references

Step 2: Start and Stop methods

- You already have these – the legacy RC scripts
- Remove links from the rc<n>.d directories
- Leave them in /etc/init.d or move to some place such as /etc/opt/svc/method
- Modify start, stop, and possibly restart methods to point to the correct RC scripts
  - Use %m in exec name to pass “start”, “stop”, and “restart”
  - exec=”:kill -<signal>” is simple shortcut to replace pkills
  - Use timeout_seconds if things take a long time to complete
- See smf_method(5) for more information

Our new manifest – so far

```
<service_bundle type='manifest' name='MySQL'>

<service  name=”application/mysql'  type='service' version='1'>

<create_default_instance enabled='true' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<exec_method type='method' name='start' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
<exec_method type='method' name='restart' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>

<property_group name='startd' type='framework'>
          <propval name='duration' type='astring' value='transient' />
</property_group>

<stability value='Unstable' />

<template>
    <common_name>
        <loctext xml:lang='C'>  MySQL Database </loctext>
    </common_name>

    <documentation>
        <manpage title='mysql' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqld' section='1' manpath='/usr/sfw/man' />
        <manpage title='mysqladmin' section='1' manpath='/usr/sfw/man' />
    </documentation>
</template>
</service>
</service_bundle>
```

Start and Stop methods

Part 3: Dependencies

- This is where things begin to get complicated
- A dependency graph is invaluable
- Start with the obvious ones
  - Run level
  - Availability of file systems
  - Configuration files
- Then add dependencies on other services
  - Need to know the error boundaries – what happens to this service when other services go offline

Example of a run level dependency

Example of a Run Level 3 dependency

```
<dependency
        name='multi-user-server'
        type='service'
        grouping='require_all'
        restart_on='none'>
  <service_fmri value='svc:/milestone/multi-user-server' />
  </dependency>
```

Use svc:/milestone/multi-user for Run Level 2 services

Local filesystems dependency

```
<dependency
        name='filesystem'
        grouping='require_all'
        restart_on='none'
        type='service'>
         <service_fmri value='svc:/system/filesystem/local'/>
</dependency>
```

Be careful with this one: an error from mountall(1M) might prevent service from starting

Local filesystems dependency (cont)

Other file system services

<service_fmri value='svc:/system/filesystem/root'/>
<service_fmri value='svc:/system/filesystem/usr'/>
<service_fmri value='svc:/system/filesystem/minimal'/>

These are just services, so use SMF to find the mount scripts to see what's really happening!

```
# svcprop -p start/exec minimal
/lib/svc/method/fs-minimal
```

Dependency on a configuration file

```
<dependency
        name='database_configuration_file'
        type='path'
        grouping='require_all'
        restart_on='refresh'>
 <service_fmri value='file://localhost/var/mysql/my.cnf' />
 </dependency>
```

Extremely handy!!!! How many times have you wondered why the NFS server didn't start ??????

Config file dependency in action

- We notice that MySQL isn't starting

```
# svcs mysql
STATE          STIME    FMRI
offline        15:17:09 svc:/application/mysql:default
```

- Using SMF we can ask why

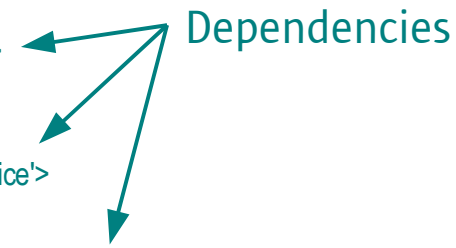
```
# svcs -l mysql
fmri         svc:/application/mysql:default
name         MySQL Database Server
enabled      true
state        offline
next_state   none
state_time   Wed Sep 07 15:17:09 2005
logfile      /var/svc/log/application-mysql:default.log
dependency   require_all/refresh file://localhost/var/mysql/my.cnf (absent)
```

MySQL dependencies

- Started at Run Level 3
- After all local file systems are mounted
- Only if there is a configuration file
  - Sample configuration files can be found in /usr/sfw/share/mysql
  - Will keep MySQL from being started in a local zone where setup process hasn't been completed
- Any others ?

Our new manifest – so far

```
<service_bundle type='manifest' name='MySQL'>

<service  name=”application/mysql'  type='service' version='1'>

<create_default_instance enabled='true' />
<single_instance />

<dependency name='multi-user-server' type='service' grouping='require_all' restart_on='none'>
        <service_fmri value='svc:/milestone/multi-user-server' />
</dependency>

<dependency  name='filesystem'  grouping='require_all'  restart_on='none'  type='service'>
        <service_fmri value='svc:/system/filesystem/local'/>
 </dependency>

 <dependency  name='database_configuration'  type='path'  grouping='require_all'  restart_on='refresh'>
        <service_fmri value='file://localhost/var/mysql/my.cnf' />
  </dependency>

<exec_method type='method' name='start' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='60'> </exec_method>
<exec_method type='method' name='stop' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
<exec_method type='method' name='restart' exec='/etc/sfw/mysql/mysql.server %m' timeout_seconds='120'> </exec_method>
```



Transient services

- Some services should not be restarted when all processes terminate
  - > Example: one time initializations

```
<property_group name='startd' type='framework'>
                <propval name='duration' type='astring'
   value='transient' />
</property_group>
```

- Include in SMF manifest or modify later using svcadm(1M)

Let's import our service

- Use svccfg or /lib/svc/method/manifest-import

```
# svcs mysql
svcs: Pattern 'mysql' doesn't match any instances
STATE          STIME    FMRI

# /lib/svc/method/manifest-import
Loaded 1 smf(5) service descriptions

# svcs mysql
STATE          STIME    FMRI
online         22:39:54 svc:/application/mysql:default

# mysqladmin status
Uptime: 1399  Threads: 1  Questions: 1  Slow queries: 0  Opens: 6
Flush tables: 1  Open tables: 0  Queries per second avg: 0.001
```

SMF in action

- Let's terminate the database daemon and see what happens

```
# svcs mysql
STATE          STIME    FMRI
online         22:39:54 svc:/application/mysql:default

# pkill mysqld

# svcs mysql
STATE          STIME    FMRI
online         22:45:45 svc:/application/mysql:default
```

Next steps

- Common:
  - Trivial: create simple service manifest, convert init scripts to service methods, minimal testing
  - Add privileges for delegated administration
  - Enable resource management
  - Full restartability: split monolithic services, each separately restartable component becomes its own service
- Advanced:
  - Customized error/restart handling: avoid service restart if fault can be internally handled

Lab #3 – Migrate an Legacy Service

Apache 1

Perform simple Apache 1 setup

```
# cd /etc/apache
# cp httpd.conf-example httpd.conf
```

Edit httpd.conf and change “Port 80” to “Port 81”

```
# /etc/init.d/apache start
```

Lauch a browser and try to connect to

http://localhost:81

Stop Apache

```
/etc/init.d/apache stop
```

Lab #3 – Migrate an Legacy Service

Apache 1

1) Migrate the Apache 1 legacy service to SMF
2) Permanently enable Apache 1 service
3) Kill all Apache 1 processes
   - What happens ?
   - Where was this event logged ?
4) What happens if we enable Apache 2 ?

Lab #3 – Migrate an Legacy Service

Apache 1

Start with Apache 2 manifest

- Change service name
- Modify start/stop methods to call /etc/init.d/apache
- Remove refresh method
- Add restart method

Add the manifest

```
# /lib/svc/method/manifest-import \
        /var/svc/manifest/network/http-apache1.xml
```

Lab #3 – Migrate an Legacy Service

Apache 1

Permanently enable Apache 1 service

1) Set enabled=true on the instance property
2) # svcadm enable apache1
3) Use svccfg to modify property

```
# svccfg
svccfg> use http:apache
svccfg> setprop general/enabled=true
svccfg> end
svcadm refresh apache1
```

Lab #3 – Migrate an Legacy Service

Apache 1

Kill all Apache 1 processes

```
# pkill httpd
```

What happens ?

svc.startd restarts the Apache 1 service

Where was this event logged ?

/var/svc/log/network-http:apache1.log

Lab #3 – Migrate an Legacy Service

Apache 1

What happens if we enable Apache 2 ?

You have 2 instances of the http service

http:apache1

http:apache2

Boot Process

In this module we will learn

- define milestones
- how milestones relate to run levels
- boot to a milestone

Boot Process

- System V RC scripts are only now run for legacy services.
- All Solaris services are started from methods in /lib/svc/method
- There are new boot milestones

  none – before any services are started

  all – all available services started
- /etc/init.d may eventually be empty

Milestones and Run Levels

SVR4 Run Level	SMF Milestone
-	none
s, S	single-user
2	multi-user
3	multi-user-server
-	All

Boot Process
run levels and milestones

- Set the default milestone
  - > **svcadm milestone -d single-user**
- Transition immediately to a milestone
  - > **svcadm milestone single-user**
- What is the current milestone ?
  - > **svcprop -p options_ovr/milestone \ system/svc/restarter:default**
  - > If property is missing then milestone is all

Boot Process-w/o services

ok **boot -m milestone=none**

login as root

Remount root filesystem as writeable

Perform required maintenance

Enable all services.

```
# svcadm milestone all
# exit
```

Boot Process-review

- SMF methods replace SysV rc scripts
- There are 5 milestones – 2 new
- Boot to a milestone instead of a run level
- Use init(1M) to transition between run levels

Lab #4 – Differences between milestones and run levels

- Boot single user
- Log in as root
- Exit the shell
- What happens ?
- Why ?
- What happens if you boot to milestone none ?

Lab #4 – Differences between milestones and run levels

- What happens ?
  - The current root shell exits
  - The system transitions to the default milestone
- Why ?
  - Consistent with previous versions of Solaris
- What else could you have done ?

  svcadm milestone all

SMF Delegated Restarters

- Provides a mechanism to handle a class of services with common startup or shutdown requirements
- Can support different methods but provides a consistent SMF interface
- The restarter's name is stored with the service.
- Example: inetd(1M)
  - > Starts services on demand
  - > Maintains additional service configuration data
- Can you think of another delegated restarter ?

inetd Managed Services

- /etc/inetd.conf converted to SMF manifest and repository on initial boot
- FMRI for converted inetd services
  - svc:/network/<servicename>:default
- inetconv(1M) adds new network services
  - - does not modify existing entries (no deletes)
  - - allows installation of third party applications that create /etc/inetd.conf entries
- Use inetadm(1M) to modify properties of inetd managed services

Viewing inetd managed services

```
% inetadm
ENABLED   STATE          FMRI
enabled   offline        svc:/application/print/rfc1179:default
disabled  disabled       svc:/network/tname:default
enabled   online         svc:/network/security/ktkt_warn:ticotsord
enabled   online         svc:/network/telnet:default
enabled   online         svc:/network/rpc/smserver:default
disabled  disabled       svc:/network/rpc/mdcomm:tcp
disabled  disabled       svc:/network/rpc/mdcomm:tcp6
enabled   online         svc:/network/rpc/gss:ticotsord
disabled  disabled       svc:/network/time:stream
enabled   online         svc:/network/nfs/rquota:ticlts
enabled   online         svc:/network/nfs/rquota:udp
enabled   online         svc:/network/ftp:default
enabled   online         svc:/network/finger:default
disabled  disabled       svc:/network/login:eklogin
disabled  disabled       svc:/network/login:klogin
disabled  disabled       svc:/network/login:rlogin
disabled  disabled       svc:/network/rexec:tcp
disabled  disabled       svc:/network/rexec:tcp6udp6
enabled   online         svc:/network/rpc-100235_1/rpc_ticotsord:ticotsord
enabled   online         svc:/network/rpc-100083_1/rpc_tcp:tcp
enabled   online         svc:/network/rpc-100068_2-5/rpc_udp:udp
enabled   online         svc:/network/fs/tcp6:default
enabled   online         svc:/network/rpc-100424_1/rpc_ticotsord:ticotsord
#
```

Managing inetd services

- Become superuser or assume a role that includes the Service Management Profile.
- List the properties for the specific service.

  # inetadm -l FMRI

- Change the property for the service.

  # inetadm -m FMRI property-name=value

Example: inetd service (telnet)

inetadm -l network/telnet

```
SCOPE    NAME=VALUE
         name="telnet"
         endpoint_type="stream"
         proto="tcp6"
         isrpc=FALSE
         wait=FALSE
         exec="/usr/sbin/in.telnetd"
         user="root"
default  bind_addr=""
default  bind_fail_max=-1
default  bind_fail_interval=-1
default  max_con_rate=-1
default  max_copies=-1
default  con_rate_offline=-1
default  failrate_cnt=40
default  failrate_interval=60
default  inherit_env=TRUE
default  tcp_trace=FALSE
default  tcp_wrappers=FALSE
```

Example: telnet (cont)

- Change the property

 # inetadm -m network/telnet tcp_trace=TRUE

 # inetadm -l network/telnet

```
SCOPE    NAME=VALUE
         name="telnet"
         endpoint_type="stream"
         proto="tcp6"
         isrpc=FALSE
         wait=FALSE
         exec="/usr/sbin/in.telnetd"
         user="root"
default  bind_addr=""
default  max_con_rate=-1
default  max_copies=-1
default  con_rate_offline=-1
default  failrate_cnt=40
default  failrate_interval=60
default  inherit_env=TRUE
default  tcp_trace=TRUE
default  tcp_wrappers=FALSE
```

Monitoring services

How to Convert inetd.conf Entries

When adding 3rd part service to inetd.conf, you will need to update the inetd smf

```
# inetconv -i filename
```

eg.

```
# inetconv -i /etc/inetd.conf
```

Basic Commands review

- Use svcs command to see info about services on system
- Use svcadm to make changes to services
- Use inetadm to view and make changes to inetd services

Troubleshooting and recovery

SMF contains a database that can be recovered if there is corruption

we will learn how to recover a corrupted database

Troubleshooting and recovery

Become root or appropriate role

Stop the svc.startd daemon

```
# pstop ‘pgrep svc.startd‘
```

kill the svc.configd daemon.

```
# pkill svc.configd
```

Make sure / is writeable

```
# mount -o rw,remount /
```

Troubleshooting and recovery

Save the current repository for debugging.

 # cp /etc/svc/repository.db /etc/svc/repository.bad

Copy the default repository.

 # cp /lib/svc/seed/global.db /etc/svc/repository.db

reboot

Now done by

/lib/svc/bin/restore_repository

Troubleshooting and recovery

```
# svcs -x
svc:/application/print/server:default (LP Print Service)
 State: disabled since Fri Oct 29 09:13:24 2004
Reason: Disabled by an administrator.
   See: http://sun.com/msg/SMF-8000-05
   See: lpsched(1M)
Impact: 2 services are not running.

svc:/network/rpc/keyserv:default (RPC Encryption Key Storage)
 State: maintenance since Fri Oct 29 09:13:39 2004
Reason: Start method failed repeatedly, last exited with status
   1.
   See: http://sun.com/msg/SMF-8000-KS
   See: keyserv(1M)
Impact: 0 services are not running.
```

Lab #5 - Boot without services and recover

1) Assume that your services database has been corrupted

   dd if=/dev/random of=/etc/svc/repository.db bs=1024 count=1000

2) A boot to the current default state will leave the system unusable

3) What do you do ?

Lab #5 - Boot without services and recover

Boot single user

(b)oot or (i)nterpreter: **b -m milestone=none**

Restore database from snapshot or seed

Import additional manifests via svccfg

- why would you do this ?

Reboot

Lab #6 – Clean Boot

```
Select (b)oot or (i)nterpreter:
SunOS Release 5.10 Version j10_70 32-bit
Copyright 1983-2004 Sun Microsystems, Inc.  All rights reserved
Use is subject to license terms.
Hostname: pandora
checking ufs filesystems
/dev/rdsk/c0d0s7: is logging

Nov 14 20:18:56 svc.startd[7]: svc:/network/rpc/keyserv:default: Method /
   usr/sbin/keyserv" failed with status 1
Nov 14 20:18:56 svc.startd[7]: svc:/network/rpc/keyserv:default: Method /
   usr/sbin/keyserv" failed with status 1
Nov 14 20:18:56 svc.startd[7]: svc:/network/rpc/keyserv:default: Method /
   usr/sbin/keyserv" failed with status 1
[svc:/network/rpc/keyserv:default failed (see 'svcs -x' for details) ]

Nov 14 20:19:02 pandora sendmail[285]: unable to qualify my own domain name
   (localhost) --using short name

Nov 14 20:19:02 pandora sendmail[286]: unable to qualify my own domain name
   (localhost) --using short name



pandora console login:
```

Lab #6 – Clean Boot

1) Figure out what services are complaining
2) Correct the problem or make it go away

Lab #6 – Clean Boot

Your boot should be nearly as clean as this

```
Select (b)oot or (i)nterpreter:
SunOS Release 5.10 Version j10_70 32-bit
Copyright 1983-2004 Sun Microsystems, Inc.  All rights reserved
Use is subject to license terms.
Hostname: pandora
checking ufs filesystems
/dev/rdsk/c0d0s7: is logging


pandora console login:
```

Lab #6 – Clean Boot

- Sendmail
  - svcs sendmail shows enabled so it was just complaining at boot time
  - Add a host.domain to your localhost in /etc/hosts
  - Or disable sendmail
- keyserv
  - svcs keyserv shows service is in maintenance mode
  - Disable keyserv or add /etc/defaultdomain

Troubleshooting common errors

- Start with svcs -x
- Make sure MySQL isn't running from a previous step
  - Service will fail to start and go into maintenance state
- Check the service log
  - Can be found using svcprop -p restarter/logfile mysql
- Check dependencies (svcs -l mysql)

Using SMF to diagnose common zone configuration problem

- Configured and installed local zone
- Boot the local zone
- Can zlogin to the local zone, but no network access
- First check svcs -a and look for uninitialized services
  - Telltale sign that SMF not started due to incomplete initialization
  - Finish sysidconfig
  - zlogin -C to complete identification process

Using SMF to diagnose common zone configuration problem

```
# zoneadm -z zone1 boot
# zlogin zone1
# svcs -a
...
offline         20:58:01 svc:/system/sysidtool:system
offline         20:58:01 svc:/milestone/sysconfig:default
...
uninitialized   20:58:01 svc:/network/rpc/gss:default
uninitialized   20:58:01 svc:/application/font/stfsloader:default
uninitialized   20:58:02 svc:/application/print/rfc1179:default
uninitialized   20:58:02 svc:/application/x11/xfs:default
...
```

Additional Resources

- OpenSolaris SMF Community
  http://opensolaris.org/os/community/smf
- Additional quickstart and developer documentation
  http://www.sun.com/bigadmin/content/selfheal/
- Solaris System Administration Guide
  http://docs.sun.com/app/docs/doc/817-1985
- Blogs:
  - http://blogs.sun.com/sch
  - http://blogs.sun.com/lianep
  - http://blogs.sun.com/jwadams
  - http://blogs.sun.com/dep
  - http://blogs.sun.com/dminer
  - http://blogs.sun.com/bobn (this workshop)

Questions ?

Thank you!

Solaris 10 Workshop
Service Management Facility

Bob Netherton
bob.netherton@sun.com
http://blogs.sun.com/bobn