

Managing Services and Service Properties by Using SMF

Objectives

After completing this lesson, you should be able to:

- Describe the advanced features of SMF
- Configure SMF services
- Troubleshoot SMF services

Job Workflow



Agenda

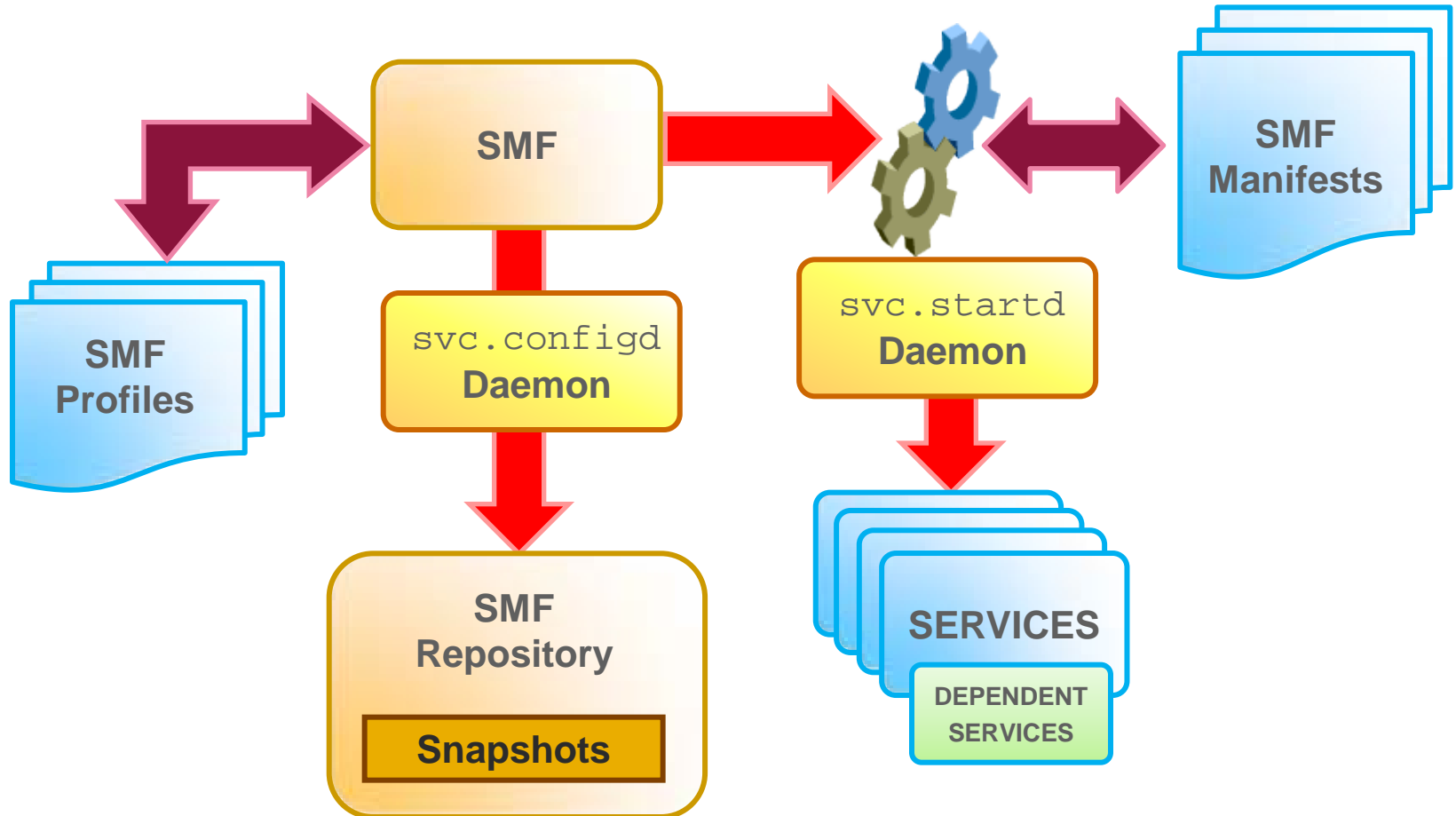
- Advanced features of SMF
- Configuring SMF services
- Troubleshooting SMF services

Importance of Service Configuration

Service configuration helps to ensure that:

- Appropriate services are enabled and running
- Existing services can be easily modified
- Downed services can be recovered and restored quickly
- New services can be created to meet emerging business needs

Advanced Features of SMF



SMF Profiles

- An SMF profile is an XML file that enables you to customize services and instances that are delivered by the system.
- Profiles delivered with the operating system include the following:

Profile	Description
<code>/etc/svc/profile/generic_open.xml</code>	Enables standard services
<code>/etc/svc/profile/generic_limited_net.xml</code>	Disables many Internet services
<code>/etc/svc/profile/ns_*.xml</code>	Enables services associated with the name service that is configured to run on the system
<code>/etc/svc/profile/platform_*.xml</code>	Enables services associated with particular hardware platforms

SMF Profile: Example

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM
    '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<!--
<header content omitted>
<service_bundle type='profile' name='generic_open'
  xmlns:xi='http://www.w3.org/2003/XInclude' >
  <!--
    Include name service profile, as set by system id tools.
  -->
  <xi:include href='file:/etc/svc/profile/name_service.xml' />

  <!--
    svc.startd(1M) services
  -->
  <service name='system/coreadm' version='1' type='service'>
    <instance name='default' enabled='true'/>
  </service>
  <service name='system/cron' version='1' type='service'>
    <instance name='default' enabled='true'/>
  </service>
```

When SMF Profiles Are Applied

- `/etc/svc/profile/generic.xml` profile:
 - Applied during the first boot after a new installation or an upgrade
 - Symbolically linked to `generic_open.xml` or `generic_limited_net.xml`
- The contents of `site.xml` in `/etc/svc/profile`:
 - Applied when the system is booted
 - Applied when the `svcadm restart manifest-import` command is run
- Profiles in `/etc/svc/profile` are applied during early manifest import.

SMF Manifests

- An SMF manifest is an XML file that describes a service and a set of instances.
- Manifests are imported to load the properties of that service and its instances into the service configuration repository.
- The preferred location for manifests is `/lib/svc/manifest`.
- Manifests are imported and upgraded by the `svc:/system/early-manifest-import:default` service during the boot process before any services start.
- The `/lib/svc/manifest/site` and `/var/svc/manifest/site` directories are reserved for site-specific use. Manifests in the site directory can be modified directly.

SMF Manifest: Example

```
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM
"/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<!--
<header and copyright content omitted>
<service_bundle type='manifest' name='SUNWzoner:zones'>
<service
    name='system/zones'
    type='service'
    version='1'>
<create_default_instance enabled='false' />

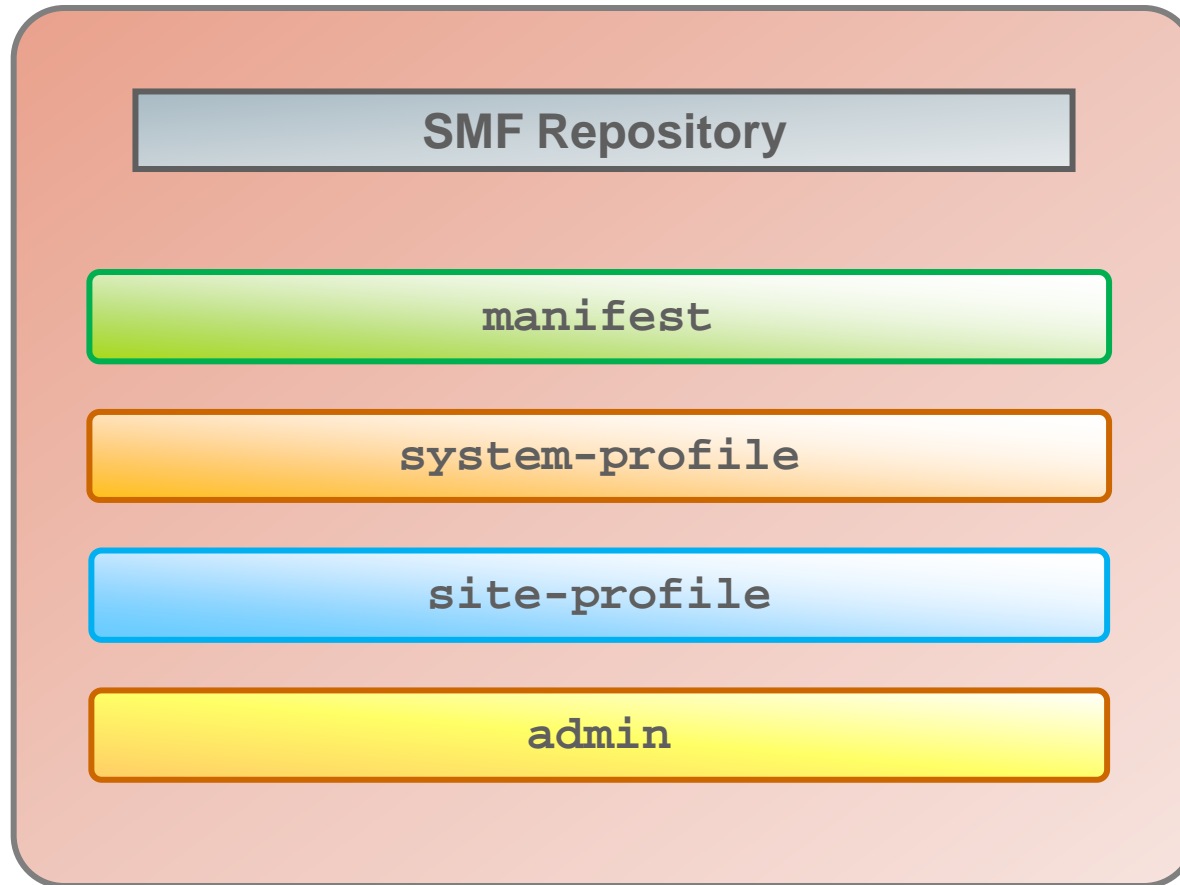
<single_instance />

---
```

Service Configuration Repository

- Stores persistent configuration information as well as SMF runtime data for services
- Is located in `/etc/svc/repository.db`
- Is managed by the `svc.configd` daemon
- Is distributed among local memory and local files
- Can be manipulated or queried only by using SMF interfaces
- Provides a consistent and persistent way to enable or disable a service
- Provides a consistent view of service state

SMF Administrative Layers



SMF Repository Backups

- SMF automatically takes the following backups:
 - **Boot backup:** Taken immediately before the first change to the repository is made during each system startup
 - **manifest_import backups:** Occur after
`svc: /system/early-manifest-import:default` or
`svc: /system/manifest-import:default` completes,
if the service imported new manifests or ran upgrade scripts
- The system maintains four copies of each type.
- Backups are stored as `/etc/svc/repository-type-YYYYMMDD_HHMMSS` for the date and time when the backup was taken.
- The repository can be restored from these backups if an error occurs.

SMF Repository Snapshots

- Snapshots are taken per service at the time when a service is successfully started.
- Standard snapshots include:
 - **initial**: Taken on the first import of the manifest
 - **running**: Taken when `svcadm refresh` is run
 - **start**: Taken at the last successful start
- The SMF service always executes with the `running` snapshot.
- Current property values for a service are incorporated into the `running` snapshot with the `svcadm refresh` command.
- Instance configurations can be viewed or reverted to in a previous snapshot by using the `svccfg` command.

Implementing Service Administration

In the next section, you learn how to:

- Create a new service and incorporate it into SMF
- Modify a service configuration
- Restore and recover a service



Quiz



The preferred location for manifests is `/lib/svc/manifest`.

- a. True
- b. False

Quiz



Which of the following profiles is used to enable standard services?

- a. `/etc/svc/profile/generic_open.xml`
- b. `/etc/svc/profile/generic_limited_net.xml`
- c. `/etc/svc/profile/ns_*.xml`
- d. `etc/svc/profile/platform_*.xml`

Quiz



Which daemon manages the service configuration repository?

- a. `svc.ipfd`
- b. `svc.configd`
- c. `svc.startd`

Quiz



The SMF service always executes with one of the following service configuration repository snapshots. Which one is it?

- a.initial
- b.running
- c.start

Agenda

- Advanced features of SMF
- **Configuring SMF services**
- Troubleshooting SMF services

Configuring SMF Services

This section covers the following topics:

- Creating a new service script
- Creating a service
- Modifying a service's manifest
- Changing an environment variable for a service
- Changing a property for an `inetd`-controlled service
- Creating and applying an SMF profile
- Changing services and their configurations by using the `netservices` command

Creating New Service Scripts

1. Determine the process for starting and stopping your service.
2. Establish a name for the service and its category.
3. Determine whether your service runs multiple instances.
4. Identify dependency relationships between this service and other services.
5. If a script is required to start and stop the process, create the script and place it in a local directory, such as `/usr/local/svc/method`.
6. Create a service manifest file for your service.
7. Incorporate the script into the SMF by using the `svccfg` utility.

Creating a Service

1. Create the script by using the following command:
`vi /usr/local/svc/method/servicename`
2. Grant the execute permission on the script so it can be executed by using the following command:
`chmod 544 /usr/local/svc/method/servicename`
3. Change directories to `/lib/svc/manifest/site` and edit the manifest `.xml` file for your new service.
4. Import the new service into SMF by using the following command:
`svcadm restart system/manifest-import`
5. Verify that the new service is available by using the `svcs servicename` command.

Creating a Service: Example

```
# mkdir -p /usr/local/svc/method/newservice
# vi /usr/local/svc/method/newservice
#!/sbin/sh
#
# ident "@(#)newservice 1.14 04/08/30 SMI"
case "$1" in
'start')
/usr/bin/newservice &
;;
'stop')
/usr/bin/pkill -x -u 0 newservice
;;
*)
echo "Usage: $0 { start | stop }"
;;
esac
exit 0
# chmod 544 /usr/local/svc/method/newservice
```

Creating a Service: Example

```
# cd /var/svc/manifest/site
# vi newservice.xml
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE service_bundle SYSTEM
    '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='manifest' name='OPTnew:newservice'>
  <service name='site/newservice' type='service' version='1'>
    <create_default_instance enabled='true'/>
    <single_instance/>
    <exec_method name='start' type='method'
      exec='/usr/local/svc/method/newservice start'
      timeout_seconds='30'>
    </exec_method>
    <exec_method name='stop' type='method' exec=':true'
      timeout_seconds='30'>
    </exec_method>
    <property_group name='startd' type='framework'>
      <propval name='duration' type='astring' value='transient'/>
    </property_group>
  </service>
</service_bundle>
```

Creating a Service: Example

```
# svccfg validate /var/svc/manifest/site/newservice.xml
# svcadm restart system/manifest-import
# svcs newservice
STATE  STIME    FMRI
online 8:43:45  svc:/site/newservice:default
```

Creating a Service: Example

```
# svcadm -v disable site/newservice
svc:/site/newservice:default disabled.
# svcs newservice
STATE      STIME      FMRI
disabled  9:11:38  svc:/site/newservice:default
# svcadm -v enable site/newservice
svc:/site/newservice:default enabled.
# svcs newservice
STATE      STIME      FMRI
online    9:11:54  svc:/site/newservice:default
#
```

Modifying a Service's Manifest

1. Modify the manifest.
2. Reimport the manifest by running `svcadm restart system/manifest-import` if in the standard location. If not in the standard location, run `svccfg import <manifest>`.
3. Importing the service will refresh it. However, a restart may be required.

Modifying a Service's Manifest: Example

```
# pfedit crmsvc.xml
# grep monitor crmsvc.xml
    <exec_method name='start' type='method'
      exec='/export/home/sstudent/smf/monitor1.crm' timeout_seconds='60' />
# svcadm restart manifest-import
# svcadm restart crmsvc
# svcadm enable crmsvc
# svcs crmsvc
```

STATE	STIME	FMRI
online	10:27:25	svc:/site/crmsvc:default

Changing an Environment Variable for a Service

1. Verify that the service is running by using `svcs FMRI`.
2. Set environment variables by using `svccfg -s FMRI setenv envar value`.
3. Refresh the service by using `svcadm refresh FMRI`.
4. Restart the service by using `svcadm restart FMRI`.
5. Verify that the change has been made by using `pargs -e `pgrep -f /usr/sbin/FMRI``.

Changing an Environment Variable for a Service: Example

```
# svcs system/cron
STATE          STIME      FMRI
online         13:02:52  svc:/system/cron:default
# svccfg -s system/cron:default setenv UMEM_DEBUG default
# svccfg -s system/cron:default setenv LD_PRELOAD libumem.so
# svcadm refresh system/cron
# svcadm restart system/cron
# pargs -e `pgrep -f /usr/sbin/cron`
2479:          /usr/sbin/cron
envp[0]: LOGNAME=root
envp[1]: _=*2476*/usr/sbin/cron
envp[2]: LANG=
envp[3]: LC_ALL=en_US.UTF-8
envp[4]: LC_COLLATE=
envp[5]: LC_CTYPE=
envp[6]: LC_MESSAGES=
envp[7]: LC_MONETARY=
envp[8]: LC_NUMERIC=
envp[9]: LC_TIME=
envp[10]: LD_PRELOAD=libumem.so
envp[11]: PATH=/usr/sbin:/usr/bin
envp[12]: PWD=/root
envp[13]: SHLVL=2
envp[14]: SMF_FMRI=svc:/system/cron:default
envp[15]: SMF_METHOD=start
envp[16]: SMF_RESTARTER=svc:/system/svc/restarter:default
envp[17]: SMF_ZONENAME=global
envp[18]: TZ=localtime
envp[19]: UMEM_DEBUG=default
envp[20]: A__z="*SHLVL#
```

Changing a Property for an `inetd`-Controlled Service

1. List the properties for the specific service by using `inetadm -l FMRI`.
2. Change the property for the service by using `inetadm -m FMRI property-name=value`.
3. Verify that the property has changed by using `inetadm -l FMRI`.
4. Confirm that the change has taken effect.

Changing a Property for an `inetd`-Controlled Service: Example

```
# inetadm -l svc:/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
default    connection_backlog=10
default    tcp_keepalive=FALSE
```

Changing a Property for an `inetd`-Controlled Service: Example

```
# inetadm -m telnet tcp_trace=TRUE
# inetadm -l svc:/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
           tcp_trace=FALSE
default    tcp_wrappers=FALSE
default    connection_backlog=10
default    tcp_keepalive=FALSE
```

Changing a Property for an `inetd`-Controlled Service: Example

```
# tail -1 /var/adm/messages
Dec 15 08:04:39 client1 inetd[655]: [ID 317013 daemon.notice]
      telnet[2390] from 192.168.0.100 34098
# grep /var/adm/messages /etc/syslog.conf
*.err;kern.debug;daemon.notice;mail.crit /var/adm/messages
```

Creating and Applying an SMF Profile

1. Create a profile by using `svccfg extract> profile.xml`.
2. Edit the `profile.xml` file to make changes:
 - a. Change the name of the profile in the `service_bundle` declaration.
 - b. Remove services that should not be managed by this profile.
 - c. Add services that should be managed by this profile.
 - d. If necessary, change the enabled flag for selected services.
3. When necessary, apply the new profile by using `svccfg apply profile.xml`.

Creating and Applying an SMF Profile: Example

```
# svccfg extract > profile.xml
# vi profile.xml
# cat profile.xml
...
<service_bundle type='profile' name='profile'
  xmlns:xi='http://www.w3.org/2003/XInclude'
  ...
  <service name='network/ldap/client' version='1' type='service'>
    <instance name='default' enabled='true' />
  </service>
  ...
  <service name='network/smtp' version='1' type='service'>
    <instance name='sendmail' enabled='false' />
  </service>
  ...
# svccfg apply profile.xml
```

Changing Services and Their Configurations by Using the `netservices` Command

Run the `netservices` command to select either open (traditional) or limited network exposure.

- For open (traditional) network exposure, run `/usr/sbin/netservices open`.
- For limited network exposure, run `/usr/sbin/netservices limited`.

Modifying a Property Value of an SMF Service

1. Identify the SMF service whose property value that you want to modify.
2. List the properties of the SMF service by using the `svccprop FMRI` command
3. Identify the property of the SMF service that you want to modify.
4. Modify the property value by using the `svccfg -s FMRI setprop property=value` command.
5. Refresh the SMF service instance so that the changes take effect.

Modifying a Property Value of an SMF Service: Example

In the following example, the host name of the system is modified:

```
root@s11-server1:~# svccprop svc:/system/identity:node
config/enable_mapping boolean true
config/ignore_dhcp_hostname boolean false
config/loopback astring ""
config/nodename astring s11-server1
...
...
root@s11-server1:~# svccfg -s identity:node describe config/nodename
config/nodename astring solaris
    Network name of the computer
root@s11-server1:~# svccfg -s svc:/system/identity:node \
setprop config/nodename=solaris11
root@s11-server1:~# svcadm refresh svc:/system/identity:node
root@s11-server1:~# Hostname:solaris11
Aug 5 10:33:12 s11-server1 rpcbind: rpcbind terminating on signal.
root@s11-server1:~# svcadm restart identity:node
root@s11-server1:~# Hostname:solaris11
root@s11-server1:~# exit
logout
oracle@s11-server1:~$ exit
logout
solaris11 console login: oracle
Password: oracle1
oracle@solaris11:~$
```

Quiz



Which of the following commands should you use to validate the manifest file?

- a. `svcs`
- b. `svcadm`
- c. `svccfg`

Quiz



If you want to display a list of services that are enabled or disabled on the current system, which of the following commands should you use?

- a. `svccfg list`
- b. `svccfg extract`
- c. `svccfg export`

Practice 2-1 and Practice 2-2 Overview: Configuring SMF Services and Working with Service Profiles

These practices cover the following topics:

- Creating and exporting a service
- Modifying a service configuration
- Changing an environment variable for a service
- Changing a property for a service controlled by `inetd`
- Creating an SMF profile
- Applying an SMF profile
- Changing the services and their configuration by using the `netservices` command

Agenda

- Advanced features of SMF
- Configuring SMF services
- Troubleshooting SMF services

Troubleshooting SMF Services

This section covers the following topics:

- Debugging a service that is not starting
- Restoring a service that is in maintenance state
- Reverting to an SMF snapshot
- Repairing a corrupt repository
- Debugging services during a system boot
- Addressing `system/filesystem/local:default` service failures during a boot

Debugging a Service That Is Not Starting

1. Request information about the hung service by using `svcs -xv servicename`.
2. Enable the service by using `svcadm enable serviceinstance`.
3. Verify that the service is online by using `svcs -a servicename`.

Debugging a Service That Is Not Starting: Example

```
# svcs -xv
svc:/network/sendmail-client:default (sendmail SMTP client queue runner)
  State: disabled since September  4, 2015 06:41:18 AM IST
  Reason: Disabled by an administrator.
    See: http://support.oracle.com/msg/SMF-8000-05
    See: man -M /usr/share/man -s 1M sendmail
  Impact: 1 dependent service is not running:
    svc:/system/fm/smtp-notify:default

svc:/system/ocm:default (Oracle Configuration Manager (OCM))
  State: maintenance since September  4, 2015 06:42:16 AM IST
  Reason: Method failed.
    See: http://support.oracle.com/msg/SMF-8000-8Q
    See: http://download.oracle.com/docs/d/E23562\_01/doc.1035/e22050/toc.htm
    See: man -M /usr/share/man -s 1M configCCR
    See: /var/svc/log/system-ocm:default.log
  Impact: This service is not running.
# svcadm enable svc:/network/sendmail-client:default
# svcadm disable svc:/system/ocm:default
# svcs ocm
STATE          STIME          FMRI
disabled        5:09:54        svc:/system/ocm:default
# svcs sendmail-client
STATE          STIME          FMRI
online          5:29:17        svc:/network/sendmail-client:default
# svcs -xv
```

Restoring a Service That Is in Maintenance State

1. Determine why the service is in maintenance by using `svcs -x FMRI`.
2. Determine whether processes that are dependent on the service have stopped by using `svcs -o CTID FMRI`.
3. (Optional) Kill any remaining processes as required by using `pkill -9 PID` for all process that are displayed by the `svcs` command.
4. Restore the service by using `svcadm clear FMRI`.

Restoring a Service That Is in Maintenance State: Example

```
# svcs -x svc:/system/ocm:default
svc:/system/ocm:default (Oracle Configuration Manager (OCM))
  State: maintenance since September 12, 2015 06:39:11 AM IST
Reason: Method failed.
  See: http://support.oracle.com/msg/SMF-8000-8Q
  See:
http://download.oracle.com/docs/cd/E23562\_01/doc.1035/e22050/toc.htm
  See: configCCR(1M)
  See: /var/svc/log/system-ocm:default.log
Impact: This service is not running.
# svcs -o CTID svc:/system/ocm:default
CTID
    218
You have new mail in /var/mail/root
# pkill -9 218
# svcadm clear svc:/system/ocm:default
# svcs svc:/system/ocm:default
STATE          STIME          FMRI
disabled        6:40:16  svc:/system/ocm:default
```

Reverting to an SMF Snapshot

1. Run the `svccfg` command.
 - a. Select the service instance that you want to fix.
 - b. Generate a list of available snapshots by using `listsnap`.
 - c. Select to revert to the `start` snapshot by using `revert start`.
 - d. Quit `svccfg` by using `quit`.
2. Update the information in the service configuration repository by using `svcadm refresh FMRI`.
3. Restart the service instance by using `svcadm restart FMRI`.

Reverting to an SMF Snapshot: Example

```
# svccfg
svc:> select system/console-login:default
svc:/system/console-login:default> listsnap
previous
running
start
svc:/system/console-login:default> revert start
svc:/system/console-login:default> quit
# svcadm refresh system/console-login:default
# svcadm restart system/console-login:default
# svcs console-login:default
STATE      STIME      FMRI
online    18:15:32  svc:/system/console-login:default
```

Configuration Repository Failed Integrity Check Process

A message is sent to the console if the integrity check fails:

```
<MESSAGE DISPLAYED BY SMF>
```

```
svc.configd: smf(5) database integrity check of:
```

```
/etc/svc/repository.db
```

failed. The database might be damaged or a media error might have prevented it from being verified. Additional information useful to your service provider is in:

```
/etc/svc/volatile/db_errors
```

The system will not be able to boot until you have restored a working database. `svc.startd(1M)` will provide a `sulogin(1M)` prompt for recovery purposes. The command:

```
/lib/svc/bin/restore_repository
```

can be run to restore a backup version of your repository. See <http://sun.com/msg/SMF-8000-MY> for more information.

Repairing a Corrupt Repository

1. Enter the `root` password at the `sulogin` prompt.
2. Run the following command:
`/lib/svc/bin/restore_repository`
3. Enter the appropriate response.
4. Enter `yes` to remedy the fault.

Repairing a Corrupt Repository: Example

```
# cd /lib/svc/bin
#:/lib/svc/bin# ./restore_repository
```

<output omitted>

The following backups of /etc/svc/repository.db exist, from oldest to newest:

```
manifest_import-20111215_035411
boot-20151214_124026
boot-20151215_150206
```

Please enter either a specific backup repository from the above list to restore it, or one of the following choices:

CHOICE	ACTION
-----	-----
boot	restore the most recent post-boot backup
manifest_import	restore the most recent manifest_import backup
-seed-	restore the initial starting repository (All customizations will be lost, including those made by the install/upgrade process.)
-quit-	cancel script and quit

Enter response [boot]: **boot-20151215_150206**

Here, you are reverting to the service repository version created on December 15, 2015

Repairing a Corrupt Repository: Example

<output continued from previous page>

...

...

After confirmation, the following steps will be taken:

svc.startd(1M) and svc.configd(1M) will be quiesced, if running.

/etc/svc/repository.db

-- renamed --> /etc/svc/repository.db_old_20151215_060922

/etc/svc/repository-boot-20151215_150206

-- copied --> /etc/svc/repository.db

and the system will be rebooted with reboot(1M).

Proceed [yes/no]? **yes**

Debugging the Services During a System Boot

1. Log in to the system as `root`.
2. Enable all services by using `svcadm milestone all`.
3. Determine where the boot process is hanging:
 - a. Run `svcs -a` to determine which services are not running.
 - b. Look for error messages in the log files in `/var/svc/log`.
4. After fixing the problems, verify that all services have started:
 - a. Verify that all needed services are online by using `svcs -x`.
 - b. Verify that the `console-login` service dependencies are satisfied by using `svcs -l system/console-login:default`.
5. Continue the normal booting process.

Addressing system/filesystem/local:default Service Failures During a Boot

1. Modify the system/console-login service by using `svccfg -s svc:/system/console-login`:

```
svc:/system/console-login> addpg site,filesystem-local dependency
svc:/system/console-login> setprop site,filesystem-local/entities = fmri:
  svc:/system/filesystem/local
svc:/system/console-login> setprop site,filesystem-local/grouping = astring:
  require_all
svc:/system/console-login> setprop site,filesystem-local/restart_on = astring: none
svc:/system/console-login> setprop site,filesystem-local/type = astring: service
svc:/system/console-login> end
```

2. Refresh the service by using `svcadm refresh console-login`.

Practice 2-3 Overview: Restoring and Recovering a Service

This practice covers the following topics:

- Restoring a service in the `maintenance` state
- Reverting to a previous SMF snapshot
- Repairing a corrupt repository
- Debugging a service that is not starting

Summary

In this lesson, you should have learned how to:

- Describe the advanced features of SMF
- Configure SMF services
- Troubleshoot SMF services