Solaris 10
SMF (Service Management Facility)

Release 6.0: 26.10.2011/bk

Hinweise, Tipps, Anregungen, Kritik, Verbesserungsvorschläge bitte an den
Autor: Helmut.Birk@edv-birk.de Homepage: http://www.edv-birk.de

INHALT

Modul 1 Service Management Facility

- **Überblick**

1-1 Allgemeines

Die Service Management Facility (SMF) ersetzt das klassische Run-Control-Konzept von Sun Solaris.
Die wesentlichen Features der SMF:
- Persistenter Start und Stop von Systemdiensten
- Berücksichtigung von Abhängigkeiten
- Paralleler Start bzw. Stop von Servicen beim Booten
- Faultmonitoring der Systemdienste (Restart)
- einheitliches CLI-Administrationsinterface
Als Service versteht die SMF nicht nur ein Systemdienst, sondern auch den Zustand eines Gerätes, wie z.B. einer Netzwerkkarte oder eines Filesystemes.

Alle Services und dessen Eigenschaften werden in einer binären SQL Lite Datenbank gehalten (/etc/svc/repository.db)

Nicht alle Solaris Systemdienste sind z.Zt. in die SMF überführt.
Diese erscheinen beim Listing als **"legacy_run"** und werden wie bisher beim Booten aufgrund der Konfig. in den **/etc/rc#.d** Verzeichnissen gestartet bzw. beim Shutdown gestoppt.
Somit sind für diese Dienste auch weiterhin die Manipulation mittels
 # /etc/init.d/<dienst> start | stop möglich.
Modulanfang

1-2 SMF Kommandos und Daemonen

/lib/svc/bin/svc.configd -> smf repository daemon
/lib/svc/bin/svc.startd -> Master Restarter
/usr/lib/inet/inetd -> smf repository Daemon

svcs
 # svcs -> Ausgabe aller aktiven Services

 # svcs -a -> dto., incl. der disabled Services
 # svcs -xv -> Ausgabe "nicht laufende Prozesse"
 # svcs -v -> verbose - incl. "Contract ID's" und "Next state" Feld
 # svcs -d system/cron -> von wem ist dieser Service abhängig
 # svcs -D system/cron -> wer ist von diesem Service abhängig
 # svcs -l system/cron -> ausführliche Infos zu diesem Service
 # svcs -x system/cron -> Infos zum Service Status

svcadm
 # svcadm -v disable cron -> dauerhaftes / persistentes Deaktivieren
 # svcadm -v disable -t cron -> temporäres Deaktivieren
 # svcadm -v enable cron -> dauerhaftes / persistentes Aktivieren
Modulanfang

1-3 SMF Service

Ein SMF-Service wird über ein Service-Manifest definiert. Dieses ist ein XML File und wird beim booten oder durch das Kommando svccfg import <xml-File> in das Repository übernommen. Das Format für das Service-Manifest ist im File /usr/share/lib/xml/dtd/service_bundle.dtd.1 festgelegt.

Das Manifest ist in Kategorien unterteilt und liegt unter /var/svc/manifest :

- application
- milestone
- platform
- system
- device
- network
- site für eigene Services

Um einen eigenen Service zu integrieren sollte ein vorhandenes Manifest kopiert und abgewandelt werden.
Beispiele :

- system/utmp ist ein einfacher standalone daemon,
- system/coreadm ist ein einfacher transient Service
- network/telnet ist ein inetd(1M)-based Service.

Hinweise und Erklärungen zu einigen Einträgen in der XML-Datei:
Ein Service kann abhängig sein von anderen Services oder von Files und Directories, File(type=path) und Service type=(service)
Beispiele :

```
<dependency name='mysql_paths'
grouping='require_all'
restart_on='none'
type='path'>
<service_fmri value='file://localhost/usr/sfw/sbin/mysqld' />
</dependency>

<dependency name='mysql_network'
grouping='require_all'
restart_on='error'
```

type='service'>
<service_fmri value='svc:/network/service' />
<service_fmri value='svc:/network/physical:default'/>
</dependency>

Ein Service kann eine Abhängigkeit auf einen anderen Service legen :
Beispiele :
<dependent name="mysql_multi-user"
grouping="optional_all"
restart_on="none">
<service_fmri value='svc:/milestone/multi-user' />
</dependent>

grouping :
require_all – Alle in der Gruppe müssen online oder degraded sein bevor die Abhängigkeit erfüllt ist.
require_any – Irgendeiner in der Gruppe muss online oder degraded sein, bevor die Abhängigkeit erfüllt ist.
optional_all – Sind die Services enabled, müssen sie online oder degraded sein, bevor die Abhängigkeit erfüllt ist.
exclude_all – Sind die Services enabled und online oder degraded, darf die Abhängigkeit nicht starten.

Restart:
none – Die dependency wird nur für den start ausgewertet
error - Restart wenn die dependency auf einen Fehler trifft (z.Bsp. core dump)
restart – Wenn die dependency neu startet, wird auch dieser Service neu gestartet
refresh – Wenn die dependency refreshed wird (Konfigurationsänderung), oder restartet wird auch dieser Service neu gestartet.

Service Modelle:
Der svc.startd ist der process-based restarter. Es gibt 3 Modelle für die Services

- *Transient* -Diese Services sind nur kurzlebig und haben Konfigurationsaufgaben oder laden starten Kernel-Aufgaben (consadm.xml,dumpadm.xml u.s.w.)
- *Wait* – Diese Services warten auf 'child Prozesse' und werden restarted wenn sie sterben. (console-login.xml,sac.xml u.s.w)
- *Contract* – Diese Services sind die standard system daemons. Sie können weitere Services starten und werden über contract-ids überwacht

Das default Service Modell ist *contract.*

Beispiele für transient und wait:
<property_group name='startd' type='framework'>
<propval name='duration' type='astring' value='**transient**' />
</property_group>
<property_group name='startd' type='framework'>
<propval name='duration' type='astring' value='**child**' />
</property_group>

Fehlt der Wert, gehört der Service zum Default Modell contract

Keywords der Methoden: **:true** -> gibt nur "success" an den restarter
:kill -> killed alle Prozesse, die vom Service

gestartet worden über die Contract-IDs

Eigenen Service integrieren :
Generelle Vorgehensweise :
- Neues Script zum starten und stoppen des Dienstes nach /lib/svc/method kopieren bzw. erstellen
- das xml-File für den neuen Dienst erstellen und nach /var/svc/manifest/site kopieren.
- importieren des xml-Files, Service ist jetzt benutzbar.
- abschliessender Test nach erfolgtem reboot

Beispiel für die MySQL Datenbank:
 # vi /lib/svc/method/mysql
 # chmod +x /lib/svc/method/mysql
 # vi /var/svc/manifest/site/mysql.xml

 # svccfg import /var/svc/manifest/site/mysql.xml
 # svcs -a | grep mysql
 # svcadm -v enable svc:/site/mysqld:default

Beispiel: Löschen eines Services:
 # svccfg delete svc:/site/mysqld:default
 # rm /var/svc/manifest/site/mysql.xml
Achtung: Wird die xml-Datei nicht gelöscht, wird der Service beim nächsten Booten automatisch wieder eingepflegt !

Mit dem **svccfg** Kommando ist es möglich direkt Änderungen in der repository.db durchzuführen und vorhergehende Zustände über **listsnap** wieder herzustellen.
 # svccfg
svc:> list
svc:> select <service>
svc:> listsnap
svc:> listprop

 # svcprop svc:/network/nfs/server:default -> Anzeige der Properties, ohne svccfg zu benützen
Modulanfang

1-4 SMF Milestones

Neuen 'default-milestone' beim booten setzen durch Option -d (entspricht dem alten initdefault Eintrag in der /etc/inittab)
Ohne die Option -d wird der milestone nur temporär gewechselt,
mögliche Einträge sind : all,none,multi-user:default,multi-user-server:default
Beim Booten ist es auch möglich den milestone mitzugeben :
ok boot -m milestone=none

 # svcadm -v milestone -d multi-user:default
 # svcadm -v milestone -d all

ok boot -m milestone=none
ok boot -m debug
 # svcadm milestone all
Modulanfang

1-5 SMF Desaster Recovery

Folgende Zustände könnten auftreten, die ein Booten der Maschine ausschliesst:
- Die /etc/svc/repository.db ist korrupt oder fehlerhaft
- Die Sicherungs-Boot-Repositories sind defekt
- das ganze /etc/svc Verzeichnis ist leer.

Beim Booten meldet das System, dass der "integrity check of ...repository.db ... failed", verbleibt im Maintenance Mode und fordert das root-Passwort an.
Troubleshooting:
 # /lib/svc/method/fs-root
 # /lib/svc/method/fs-usr
 # /lib/svc/bin/restore_repository
-> danach Auswahl: einer repository, z.B. "Boot" -> nimmt die jüngste Kopie

Alternativ:
 # mount -o remount /
 # mount -o remount /usr
 # mv /etc/svc/<sicherungsdatei> /etc/svc/repositiory.db

Existieren keine Backup-Boot-files mehr, kann man eine **Default repository** aufsetzen, welche dann aus **/lib/svc/seed/global.db** kopiert wird.
....
....
 # /lib/svc/bin/restore_repository
-> danach Auswahl **-seed-**
Modulanfang

1-6 inetd Daemon

Der inetd Daemon ist weiterhin für die Internet Services zuständig.
Allerdings ist inetd jetzt sowohl ein smf-service (s**vc:/network/inetd:default**),
als auch ein sogenannter delegated restarter.
Neue Kommandos:
 # inetadm
 # inetadm -e finger -> persistentes Aktivieren
 # inetadm -d finger -> persistentes Deaktivieren
Alternativ können auch die Kommandos svcadm benützt werden.
Ein temporäres Deaktivieren ist zudem nur mit svcadm möglich:
 # svcadm -v disable -t finger
 # inetadm -l telnet -> Anzeige aller Eigenschaften

inetd liest die Konfiguration der zu betreuenden Dienste nicht mehr aus der **/etc/inetd.conf** Datei aus.
Allerdings wird die Datei zum Importieren eines neuen inetd Dienstes wie folgt benützt:
Beispiel: SWAT zur Samba-Administration
 # vi /etc/services
swat 901/tcp
:wq
 # vi /etc/inetd.conf
swat stream tcp nowait root /usr/sfw/sbin/swat swat
:wq
 # inetconv
Modulanfang

1-7 Wichtige Pfade

/lib/svc/seed/global.db (Default repository / wird beim 1. boot angelegt)
/etc/svc/ -> repository.db und dessen Sicherungs-Kopien
/etc/svc/volatile -> Log Einträge vom jüngsten Boot
/var/svc/log -> alle Log Einträge der Services
/var/svc/manifest -> Kategorien der Services
/lib/svc/method -> Methoden der Systemdienste

Modulanfang
Dokumentanfang