Thursday, February 28. 2008

Solaris Features: Service Management Facility - Part 4: Developing for SMF

Okay, now you know how to do basic tasks. But how to use SMF for your your own applications. I will use openvpn as an example. A good source for this program is blastwave. Please install the package tun and openvpn

I want to show a running example, thus we have to do some work. It will be just a simple static shared key configuration, as this is a SMF tutorial, not one for OpenVPN. We will use theoden and gandalf again. gandalf will be the server. theoden the client.10.211.55.201 gandalf
10.211.55.200 theoden

Preparing the server
Okay ... this is just a short configuration for an working OpenVPN server.

```
# mkdir /etc/openvpn
# cd /etc/openvpn
# openvpn --genkey --secret static.key
# openvpn --dev tun --ifconfig 192.16.1.1 172.16.1.2 --secret static.key --daemon
# scp static.key jmoekamp@10.211.55.200:/tmp/static.key
```

Now just leave this terminal this way.

Preparing the client
Okay, we need some basic configurations to get the client side of OpenVPN working, even when it´s unter control of the SMF # mkdir /etc/openvpn

```
# mv /tmp/static.key /etc/openvpn
# cd /etc/openvpn/
# ls -l
total 2
-rw-------   1 jmoekamp other        636 Feb 27 16:11 static.key
# chown -R root:root /etc/openvpn
# chmod 600 static.key
```

Before working with SMF itself
At first i remove this stinking init.d links. We don´t need them anymore:

```
# rm /etc/rc0.d/K16openvpn
# rm /etc/rc1.d/K16openvpn
# rm /etc/rc2.d/S60openvpn
# rm /etc/rcS.d/K16openvpn
```

Okay, and now let´s hack the startup script....? Wrong! SMF can do many task for you, but this needs careful planing. You should answer yourself some questions:

1. What variables make a generic description of a service to a specific server?
2. How do i start the process? How do i stop them? How can i force the process to reload it´s config?
3. Which services are my dependencies? Which services depend on my new service?
4. How should the service react in the case of a failed dependency?
5. What should happen in the case of a failure in the new service.

Okay, let´s answer this questions for our OpenVPN service. The variables for our OpenVPN client are the hostname of the remote hosts and the local and remote IP of the VPN tunnel . Besides of this the filename of the secret key and the tunnel device may differ, thus it would be nice to keep them configurable.

Starting openvpn is easy. We have just to start the openvpn daemon with some command line parameters. We stop the service by killing the process. And a refresh is done via stopping and starting the service.

We clearly need the networking to use a VPN service. But networking isn´t just bringing up the networking cards. You need the name services for example. So make things easier, the service don´t check for every networking service to be up and running. We just define an dependency for the "network" milestone.

As it make no sense to connect to a server without a network it looks like a sensible choice to stop the service in case of a failed networking. Furthermore it seems a good choice to restart the service when the networking configuration has changed. Perhaps we modified the configuration of the name services and the name of the OpenVPN server resolves to a different IP.

What should happen in the case of a exiting OpenVPN daemon? Of course it should started again.

Okay, now we can start with coding the scripts and xml files.

The Manifest
Okay, as i wrote before, the manifest is the source of the configuration of a service. Thus we have to write such a manifest. The manifest is a XML file. Okay, at first we obey the gods of XML and do some definitions:# cat openvpn.xml

Okay, at first we habve to name the service:

In this example, we define some simple dependencies. As i wrote before: Without networking a VPN is quite useless, thus the OpenVPN service depends on the reached milestone.

In this part of the manifest we define the exec method to start the service. We use a script to start the service. The %m is a variable. It will be substituted with the name of the called action. In this example it would be expanded to /lib/svc/method/openvpn start. Okay, we can stop OpenVPN simply by sending a SIGTERM signal to it. Thus we can use a automagical exec method. In case you use the :kill SMF will kill all processes in the actual contract of the service. Okay, thus far we´ve only define the service. Let´s define a service. We call the instance theoden2gandalf for obvious names. The service should run with root privileges. After this we define the properties of this service instance like the remote host or the file with the secret keys.

At the end we add some further metadata to this service:

OpenVPN

I saved this xml file to my homedirectory under /export/home/jmoekamp/openvpn.xml.

The exec method´s script
General considerations
Okay, we referenced a script in the exec method. This script is really similar to a normal init.d script. But there are some important differences. As there´s no parallel startup of services in init.d most scripts for this system bringup method tend to return as quickly as possible. We have to change this behaviour.

Scripts for transient or standalone services should only return in the case of the successful execution of the complete script or when we´ve terminated the process. For services under control of the contract mechanism the script should at least wait until the processes of the service generate some meaningful error messages, but they have to exit, as SMF would consider the service startup as failed, when the script doesn´t return after

There are some general tips: When you write your own stop method don´t implement it in a way that simply kills all processes with a certain name (e.g. pkill "openvpn") this was and is really bad style, as there may be several instances of service. Just using the name to stop the processes would cause unneeded collateral damage.
It´s a good practice to include the /lib/svc/share/smf_include.sh. It defines some variables for errorcodes to ease the development of the method scripts.

Implementing a exec method script
We store configuration properties in the Service Component repository. It would be nice to use them for configuration. Thus we have to access them.

Here comes the svcprop command to help: # svcprop -p openvpn/remotehost svc:/application/network/openvpn:theoden2gandalf
gandalfWith a little bit of shell scripting we can use this properties to use them for starting our processes.

```
#!/bin/sh

. /lib/svc/share/smf_include.sh

getproparg() {
val=`svcprop -p $1 $SMF_FMRI`
[ -n "$val" ] && echo $val
}

if [ -z "$SMF_FMRI" ]; then
echo "SMF framework variables are not initialized."
exit $SMF_EXIT_ERR
fi

OPENVPNBIN='/opt/csw/sbin/openvpn'
REMOTEHOST=`getproparg openvpn/remotehost`
SECRET=`getproparg openvpn/secret`
TUN_LOCAL=`getproparg openvpn/tunnel_local_ip`
TUN_REMOTE=`getproparg openvpn/tunnel_remote_ip`
DEVICETYPE=`getproparg openvpn/tunneldevice`

if [ -z "$REMOTEHOST" ]; then
echo "openvpn/remotehost property not set"
exit $SMF_EXIT_ERR_CONFIG
fi

if [ -z "$SECRET" ]; then
echo "openvpn/secret property not set"
exit $SMF_EXIT_ERR_CONFIG
fi

if [ -z "$TUN_LOCAL" ]; then
echo "openvpn/tunnel_local_ip property not set"
exit $SMF_EXIT_ERR_CONFIG
fi

if [ -z "$TUN_REMOTE" ]; then
echo "openvpn/tunnel_remote_ip property not set"
exit $SMF_EXIT_ERR_CONFIG
fi

if [ -z "$DEVICETYPE" ]; then
echo "openvpn/tunneldevice property not set"
exit $SMF_EXIT_ERR_CONFIG
fi
```

```
case "$1" in
'start')
$OPENVPNBIN --daemon --remote $REMOTEHOST --secret $SECRET --ifconfig $TUN_LOCAL $TUN_REMOTE --dev
$DEVICETYPE
;;

'stop')
echo "not implemented"
;;

'refresh')
echo "not implemented"
;;

*)
echo $"Usage: $0 {start|refresh}"
exit 1
;;

esac
exit $SMF_EXIT_OK
```

I saved this script to my homedirectory under /export/home/jmoekamp/openvpn.

Installation of the new Service
Okay, copy the script to /lib/svc/method/:# cp openvpn /lib/svc/method
# chmod +x /lib/svc/method/openvpnAfter this step you have to import the manifest into the Service Configuration Repository:# svccfg validate /export/home/jmoekamp/openvpn.xml
# svccfg import /home/jmoekamp/openvpn.xmlTesting itLet´s test our brand new service:# ping 172.16.1.2
^CThe OpenVPN service isn´t enabled. Thus there is no tunnel. The ping doesn´t get through.
Now we enable the service and test it again.
# svcadm enable openvpn:theoden2gandalf
# ping 172.16.1.2
172.16.1.2 is alive
Voila ... SMF has started our brand new service. When we look into the list of services, we will find it:# svcs openvpn:theoden2gandalf
STATE   STIME   FMRI
online   18:39:15 svc:/application/network/openvpn:theoden2gandalfWhen we look into the process table, we will finde the according process:# /usr/ucb/ps -auxwww | grep "openvpn" | grep -v "grep"
root   1588 0.0 0.5 4488 1488 ?   S 18:39:15 0:00 /opt/csw/sbin/openvpn --daemon --remote gandalf --secret /etc/openvpn/static.key --ifconfig 172.16.1.2 172.16.1.1 --dev tunOkay, we doesn´t need the tunnel any longer after a few day,thus we disable it:# svcadm disable openvpn:theoden2gandalf
# /usr/ucb/ps -auxwww | grep "openvpn" | grep -v "grep"
# No process left.

Posted by Joerg Moellenkamp in English, Solaris at 12:31

I've just stumbled upon your SMF documentary during creation of a new service for myself.. but you (and the rest of the googlable internet) failed to answer part 5: "How to react on a failure of the service". Can I convince you to elaborate on that point a bit?

I've created several SMF services for our systems already, but they all fail in one task: they don't restart automatically when the managed daemon stops/crashes.
I have found NO documentation yet about how SMF determines whether a service is still functional, how to define which process IDs or names to monitor..
I can't even get SMF to degrade the service to maintenance state so I could react somehow externally.
What good is SMF when I still need to use inittab or cronjobs to monitor my services...
Anonymous on Jul 20 2011, 14:17

Hello Woo, I realize I'm a year too late with the answer, but still - it might help somebody...

SMF allows for several service (and failure) modes. In particular, there are "transient" services which are not monitored during

live-time and basically if the start method succeeded (returned 0) the service is considered "online". In this case a stop method can be used with its own magic to locate and stop the daemons. Transient services are usually used for system configuration (i.e. set up some kernel tuning and "exit 0").
One notable example I know (and maintain) is the "vboxsvc" project on sourceforge, where VirtualBox VM processes have to be run as transient SMF services due to VBox architectural nuances. A large part of my method script is indeed a process which remains in the background and monitors the VM's state so as to cause its restart or maintenance in case of detected failures.

If your services daemons die and are not restarted by SMF, double-check that they are not configured as "transient".

Another aspect is SMF's reaction (or configurable lack of) to the signal which may be associated with the death of monitored processes in wait/contract type services. For example, if the service daemon coredumped, its service may be put into maintenance or restarted just like upon another death of the daemon.

HTH,
//Jim Klimov
Anonymous on May 9 2012, 22:33