Monday, April 2. 2012

Solaris 11 features: nscfg

As you may have noticed many configuration tasks around name services have moved into the SMF in Solaris 11. However you don't have to use the svccfg command in order to configure them, you could still use the old files. However you can't just edit them, you have to import the data into the SMF repository. There are many reasons for this need but the ultimate one is in the start method. I will explain that later. In this article i want to explain, how nscfg can help you with with the naming service configuration of your system.

The direct methodJust as a reminder. There is a direct method to enter the configuration data of the naming services directly into the SMF.For example when you want to configure a new name server you could use the method i described some days ago:

```
root@tachikoma:/home/jmoekamp# svccfg  -s "dns/client" setprop "config/nameserver = net_address: ( 192.168.1.1 )"
root@tachikoma:/home/jmoekamp# svccfg  -s "dns/client" setprop 'config/domain = astring: ("c0t0d0s0.org")'
root@tachikoma:/home/jmoekamp# svccfg -s "name-service/switch" setprop 'config/host = astring: "files dns"'
root@tachikoma:/home/jmoekamp# svcadm refresh name-service/switch
root@tachikoma:/home/jmoekamp# svcadm enable dns/client
```

Using nscfgHowever there are people, who don't really like this. There is an additional method for the naming services by using an tool that is normaly used to import legacy files into the SMF.

The tool to do so is the nscfg command. Primarily the nscfg is used to import and export name service configuration. When you export a nameserver configuration the relevant properties into the legacy config files of services (for DNS for example writing the data into resolv.conf). When you import a nameserver configuration the configuration files are parsed (for DNS again, reading the data from the resolv.conf).

Let's assume i start with a system without any networking configuration. So i have to give my Solaris instance an ip address. You know the drill:

```
root@solaris:/home/jmoekamp# ipadm create-ip net0
root@solaris:/home/jmoekamp# ipadm create-addr -T static -a 192.168.1.241/24 net0/ipv4
root@solaris:/home/jmoekamp# route -p add default 192.168.1.1
add net default: gateway 192.168.1.1
add persistent net default: gateway 192.168.1.1
root@solaris:/home/jmoekamp# ping 192.168.1.1
192.168.1.1 is aliveOkay, in the standard config, the system is not aware of something like DNS. So we have to tell it, where the nameserver is and the it should use a nameserver.root@solaris:/home/jmoekamp# echo "nameserver 192.168.1.1" > /etc/resolv.conf
root@solaris:/home/jmoekamp# cp /etc/nsswitch.dns /etc/nsswitch.confOkay, now we use the nscfg tool. In this case i'm showing the verbose output. root@solaris:/home/jmoekamp# nscfg import -fv dns/client
importing DNS legacy...
   delete customizations.
Refresh svccfg command:   /usr/sbin/svccfg -s svc:/network/dns/client:default refresh
   loading pg...
   committing pg...
Refresh svccfg command:   /usr/sbin/svccfg -s svc:/network/dns/client:default refresh
   validating pg...
Validate service properties for:   svc:/network/dns/client
successful import.
root@solaris:/home/jmoekamp#There is a second option -f in the last example. Normally when there is already some configuration data in the SMF, the tool just exists and keeps everything untouched. It would just run and import data, if it finds resolv.conf file, but no data in the SMF. So we need to specify -f in order to force the import of the current content of the file into the SMF. A short look into the smf will disclose a 192.168.1.1 as in the file as the value of this property
root@solaris:/etc# svcprop dns/client | grep "config/nameserver"
config/nameserver net_address 192.168.1.1
```

Of couse there is still the configuration for the name service switch missing. Earlier we copied the dns variant of the nsswitch.conf to /etc/nsswitch.conf. Now we have to import the setting into SMF. Again we use nscfg.

root@solaris:/home/jmoekamp# nscfg import -f name-service/switchOkay, now we test the DNS component. Just a short nslookup.root@solaris:/home/jmoekamp# nslookup c0t0d0s0.org
Server: 192.168.1.1
Address: 192.168.1.1#53

Non-authoritative answer:
Name: c0t0d0s0.org
Address: 178.63.69.146
I changed my mind. I want to use my DNS server on 192.168.1.43. Let's us configure this.
root@solaris:/home/jmoekamp# echo "nameserver 192.168.1.43" > /etc/resolv.conf
root@solaris:/home/jmoekamp# nscfg import -f dns/client
The property in SMF has changed.
root@solaris:/etc# svcprop dns/client | grep "config/nameserver"
config/nameserver net_address 192.168.1.43
And when you do the lookup onto the hostname, you will see that the DNS server used by nslookup has changed.root@solaris:/home/jmoekamp# nslookup c0t0d0s0.org
Server: 192.168.1.43
Address: 192.168.1.43#53

Non-authoritative answer:
Name: c0t0d0s0.org
Address: 178.63.69.146

root@solaris:/home/jmoekamp#To be honest, it would have used the new server even with the nscfg run. Work of putting stuff into Solaris to gateher the data directly from SMF. However one of the reasons to generate the legacy config files is to enable a smooth migration from the legacy files to the SMF

Why all this hassleHowever: When even some components of the OS doesn't use the data in the SMF, why all this hassle with nscfg and the smf. Instead of explaining it with long words, the shorter variant is just to show an example. It's my favourite one, as it's the simplest one.
root@solaris:/etc# svcs -a | grep "sendmail"
online 9:11:46 svc:/network/smtp:sendmail
online 9:11:46 svc:/network/sendmail-client:default
root@solaris:/etc# date
Sunday, April 1, 2012 09:12:33 AM UTC
root@solaris:/etc# nscfg import -f dns/client
root@solaris:/etc# svcs -a | grep "sendmail"
online 9:12:40 svc:/network/smtp:sendmail
online 9:12:40 svc:/network/sendmail-client:default
No only processes have interdependencies, but config files and and processes as well. And changing a hostname or a domainname or even the resolver config. Depending on the configuration a change in the resolv.conf may need an restart of the sendmail. Changing the domainname may have dependencies to the NIS server. When you just change the file, nothing else changes. So you may have to trigger the refresh of the configuration files yourself. By putting the lead repository of the configuration data for some configs into the SMF a refresh and thus activiation of the new config data leads automatically to the restart of the depending services. So by moving the data into the SMF configuration data gets dependencies.

Scope of the nscfg commandThe nscfg command does not only work for dns or nsswitch, but for other naming services as well:

Service	SMF	Files
Name service switch	sv	/etc/nsswitch.conf

Name service cache (nscd)
svc:/system/name-service/cache:default
/etc/nscd.conf

DNS naming service
svc:/network/dns/client:default
/etc/resolv.conf

Shared NIS domain configuration
svc:/network/nis/domain:default
/etc/defaultdomain/var/yp/binding/$DOMAIN/*

NIS client naming service
svc:/network/nis/client:default
-

LDAP client naming service
svc:/network/ldap/client:default
/var/ldap/*

NIS server service
svc:/network/nis/server:default
-

NIS server passwd service
svc:/network/nis/passwd:default
-

NIS server xfr service
svc:/network/nis/xfr:default
-

NIS server update service
svc:/network/nis/update:default
-

Import legacy configuration files into SMF service
svc:/system/name-service/upgrade:default
-

Why you can't directly edit the config files?
So, why is the startup the ultimate reason why you have to get the configuration of the name services into the SMF? It's because of the start methods of the SMF service.

```
case "$1" in
'start')
        # Test and import if upgrade
        /usr/sbin/nscfg import -q $SMF_FMRI
        [...]
        /usr/sbin/nscfg export $SMF_FMRI
  [...]
```

;;
What happens here? At first there is an import run? However as there is no -f defined, the files on the disk are just imported when the properties aren't already configured. In this case the import will just silently do nothing. In any case the system creates a new version of the name service file afterwards. So whatever no matter what you edit into this files, you always start with a freshly generated version of the state defined by the properties in the SMF.

Do you want to learn more?docs.oracle.com: nscfg(1m)

```
 Posted by Joerg Moellenkamp in English, Solaris at 09:43
```

OK, Joerg. You won me over.

-Alex
Anonymous on Apr 2 2012, 12:44