Oracle on Solaris 10 Containers/Zones

William Sescu . Consultant . 21.07.2009

Running Oracle on Solaris 10 containers has become more and more popular. There are two major reasons, for running Oracle Single Instances in a Solaris 10 container:

- **The first one is, that it is supported by Oracle**
- **And the second one, is that hard partitioning is possible, for a capped container**

Hard partitioning means, that you can assign a certain amount of CPU's or CPU cores to a capped container and only those ones have to be licensed.

For more information about running Oracle in capped Solaris containers and the Oracle license agreement, see Metalink note: 317257.1.

Beside the reasons above, the Solaris Zones feature comes with more advantages:

- **Free, is shipped with the Solaris 10 operating system**
- **Lightweight, less then 1% performance overhead**
- **Zones are available on Sparc and the x86 platform**
- **Performance monitoring over all zones can be done from the global zones**
- **Full isolated environment. Gaining root access to a zone has no impact on other zones**
- **Patches can be applied on the global zone, and are automatically applied on all other local and non branded zones**

Before we start to create our first Solaris container, I would like to explain the difference between a container and a zone, because these words are often used interchangeable.

What is a zone?

A zone is a virtual operating system abstraction that provides a protected environment in which applications run. The applications are protected from each other to provide software fault isolation. To ease the labor of managing multiple applications and their environments, they co-exist within one operating system instance, and are usually managed as one entity.

What is a container?

A zone which also uses the operating system's resource management facility is then called a container. Many people use the two words 'zone' and 'container' interchangeably.

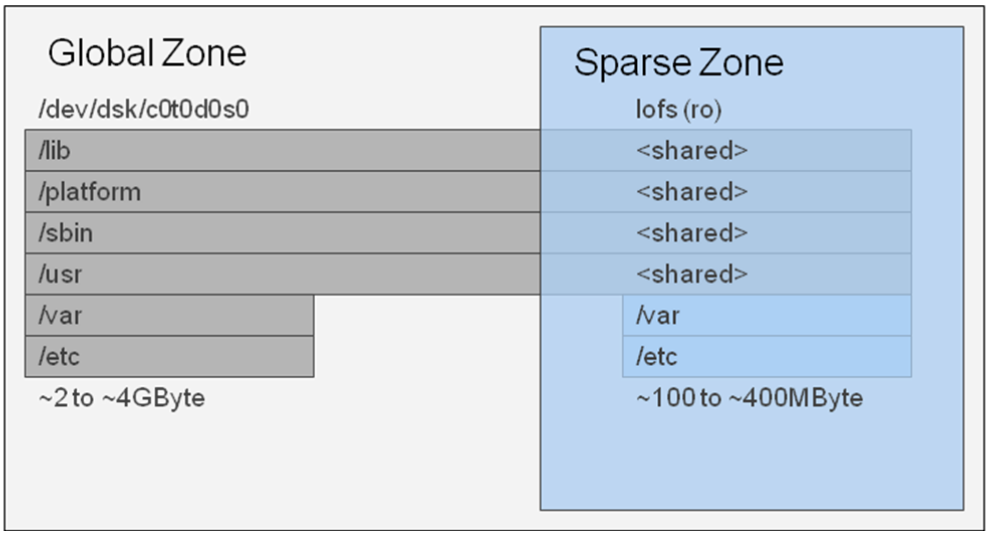
1. Sparse, Whole Rooted and Branded Zones

Solaris 10 comes with three different kinds of zones

- Sparse Zones
- Whole rooted Zones
- Branded Zones

1.1 Sparse Zones

Sparse Zones are the default, when you install a zone. The benefit of a Sparse Zone is that it is very small. Depending on your installation, it will take from 100MB to 400MB of disk space. The reason for that, is that the filesystems /usr, /lib, /sbin and /platform are shared and mounted on the local zone via the loopback filesystem. (lofs, ro)



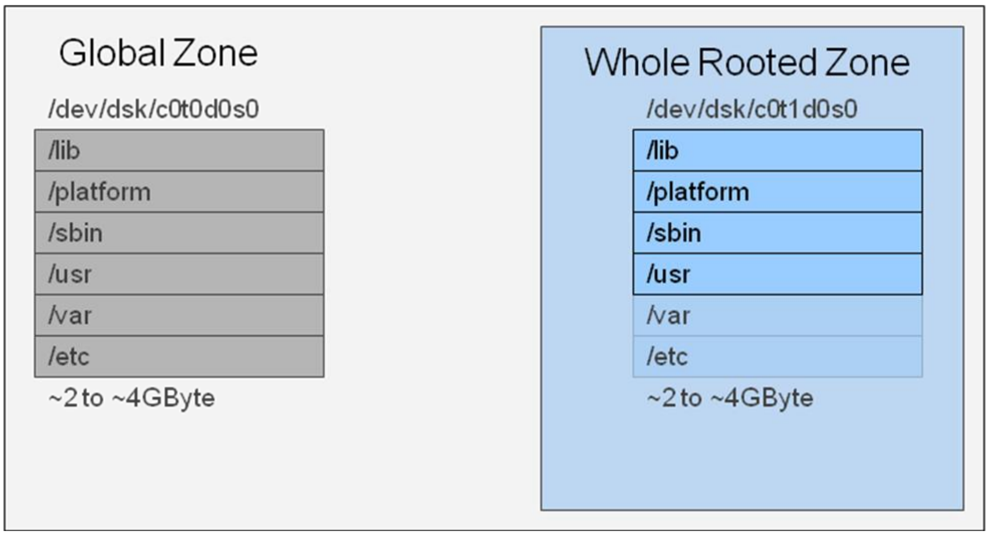
Another advantage of the Sparse Zone is that it can use the already loaded libraries by the Global Zone, and does not have to load the libraries into the memory again.

The configuration of a Zone is stored in a XML file in the /etc/zones directory. The default Zone XML file looks like the following output, where you can see the filesystems that are inherited by the Global Zone.

```
global# cat /etc/zones/SUNWdefault.xml
<zone name="default" zonepath="" autoboot="false">
  <inherited-pkg-dir directory="/lib"/>
  <inherited-pkg-dir directory="/platform"/>
  <inherited-pkg-dir directory="/sbin"/>
  <inherited-pkg-dir directory="/usr"/>
</zone>
```

1.2 Whole Rooted Zones

Whole rooted Zones do not share any filesystem and are about 2 to 4GB in size, depending on your Global Zone. The advantage of a whole rooted Zone, is that you have full flexibility. For example you can install software in the /usr/bin directory.



1.3 Branded Zones

Branded Zones are all non Solaris 10 Zones, whereby the Sparc and the x86 platform supports different kind of branded Zones.

Solaris 10 Sparc supports

- Solaris 8 Sparc branded zone
- Solaris 9 Sparc branded zone

Solaris 10 x86 supports

- Linux 32/64bit with the 2.4 Kernel

A detailed description of the supported Linux distributions can be found in the directory /usr/lib/brand/lx/distros/ on a Solaris 10 x86 systems.

```
global# ls /usr/lib/brand/lx/distros/
centos35.distro
centos36.distro
centos37.distro
centos38.distro
rhel35.distro
rhel36.distro
rhel37.distro
rhel38.distro
```

It was out of scope, to test branded zones with the Linux 2.4 Kernel, because most of the customers might not want to use the old 2.4 Linux Kernel anymore.

2. Services that a Zone can provide

There are some limitations regarding the services that can run in a Solaris Zone. E.g. Oracle RAC is not supported, and running a NFS Server can also not be started in a Solaris Zone.

Service	**Possibility to run in a Solaris 10 zone**
Oracle DB	Yes
Oracle RAC	No, not supported
Oracle Agent	On Sparc supported with >= 10.2.0.4.0 On x86 64bit supported with >= 10.2.0.5.0
NFS Server	No, because NFS Server in Solaris is built in the Kernel
NTP Client	No, the time is retrieved from the global zone. But you can have a different time zone. /etc/default/init
DNS Server	Yes
NIS, NIS+,LDAP	Yes
Sendmail	Yes
X11	Yes
DHCP Server	Yes, starting with Solaris 10 11/06
IPMP	Yes
ipfilter	Yes, with an exclusive IP instance, or on the global zone
defaultrouter	Yes, starting with Solaris 10 10/08 you can define a own default router for the zone, or you can use the one provided by the global zone in a shared ip configuration

SUN adds more and more features to the Solaris Zones, so it is advisable to take a look into the release notes, of your Solaris version.

3. Before we create our first zone

There are several issues, which got to be considered before we start to create our first zone.

Make sure that the UID's and GID's of the UNIX users match, that are created in the global zone and in the local zones. In case it is configured correctly, you will see a good `prstat -Z` output.

Configure NTP on the global zone and the local zones will get the time from there. Per default, it is not possible to start a NTP client on a local zone.

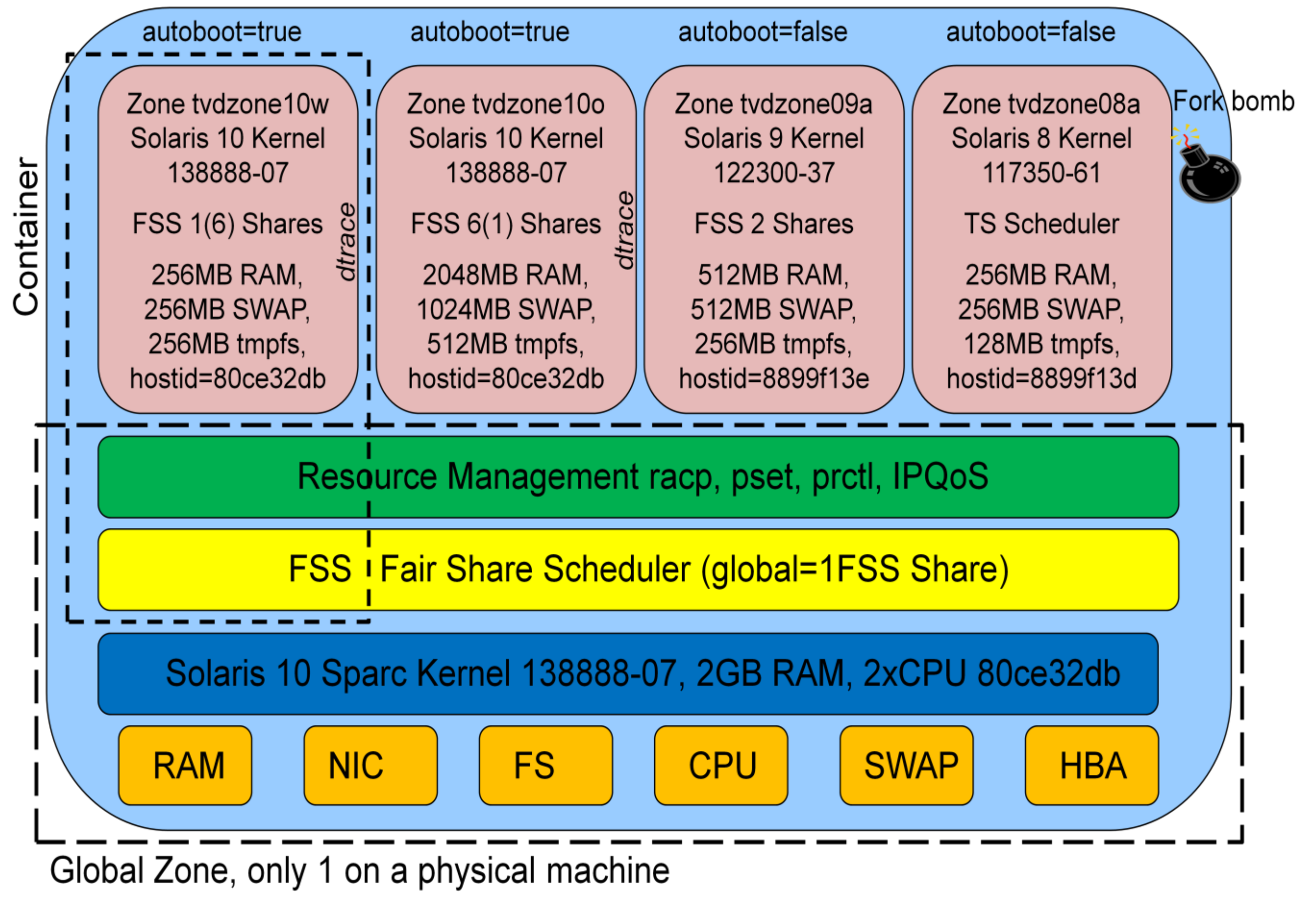
Configure IPMP on your network cards on the global zone. The VIP in the local zone will failover, whenever there is a failover on the global zone.

Configure the processor sets on the global zone, in case you need the hard portioning feature. Activate the FSS (Fair Share Scheduler) on the global zone, in case you want to use this feature.

4. My Test Environment in the Trivadis Technology Center

In our Trivadis Technology Center, I build up the environment for testing different kind of zones. I built them up on two Solaris Sparc Servers, with 2 CPU's and 2GB RAM each. Both machines had access to a shared storage. Because all the zones reside on a ZFS filesystem, I was able to move zones from machine to machine.

Let's take a look at the four zones on my test machine. I created a whole rooted zone "tvdzone10w", a sparse zone "tvdzone10o" for the Oracle database, a branded zone "tvdzone09a" with Solaris 9 and another branded zone "tvdzone08a" with Solaris 8 for the legacy applications.


autoboot=true
autoboot=true
autoboot=false
autoboot=false
Zone tvdzone10w Solaris 10 Kernel 138888-07 FSS 1(6) Shares 256MB RAM, 256MB SWAP, 256MB tmpfs, hostid=80ce32db
Zone tvdzone10o Solaris 10 Kernel 138888-07 FSS 6(1) Shares 2048MB RAM, 1024MB SWAP, 512MB tmpfs, hostid=80ce32db
Zone tvdzone09a Solaris 9 Kernel 122300-37 FSS 2 Shares 512MB RAM, 512MB SWAP, 256MB tmpfs, hostid=8899f13e
Zone tvdzone08a Solaris 8 Kernel 117350-61 TS Scheduler 256MB RAM, 256MB SWAP, 128MB tmpfs, hostid=8899f13d
dtrace
dtrace
Container
Fork bomb
Resource Management racp, pset, prctl, IPQoS
FSS Fair Share Scheduler (global=1FSS Share)
Solaris 10 Sparc Kernel 138888-07, 2GB RAM, 2xCPU 80ce32db
RAM
NIC
FS
CPU
SWAP
HBA
Global Zone, only 1 on a physical machine

As you can see on the picture, the global zone can exist only one time on a physical machine, and the global zone is responsible for taking care of the hardware. It manages the memory, the network cards, the filesystems, the cpu's and the host bus adapters.

It does not have to be that way. E.g. filesystems can be created on the local zones, and network cards can be managed on the local zones as well, depending on how you configure your zones.

On top of the hardware, we see the Solaris operating system with the Kernel 138888-07, the one I used for my tests. You might have different features available, in case you use an older one.

I activated the FSS (Fair Share Scheduler), to control the allocation of CPU resources among workloads. FSS guarantees a fair dispersion of CPU resources based on allocated shares, and this is exactly what I wanted.

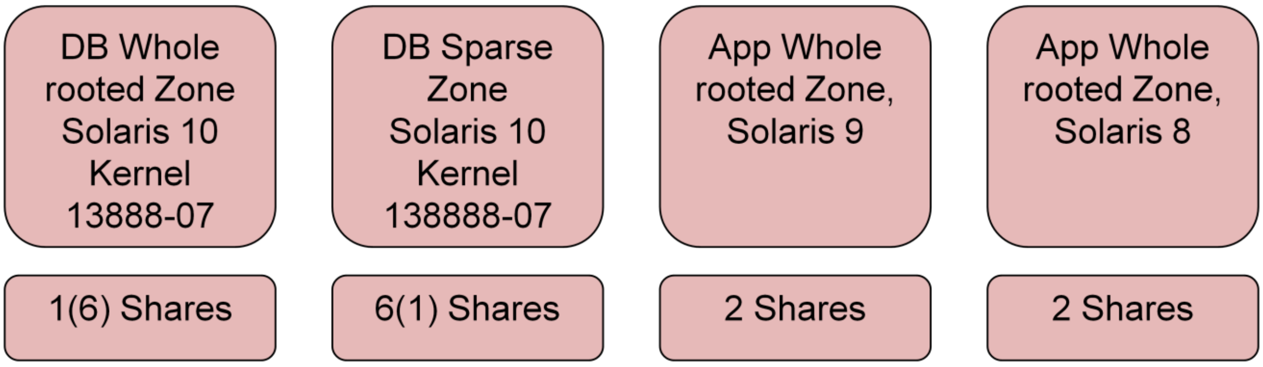
The FSS was not always available with Solaris. It was introduced with Solaris 9, and with Solaris 10 you can use it with zones.

Scheduler	Description	8	9	10
SYS	System Class	Yes	Yes	Yes
TS	Time Sharing (default)	Yes	Yes	Yes
RT	Real Time	Yes	Yes	Yes
FSS	Fair Share Scheduler	No	Yes	Yes
FX	Fixed priority	No	Yes	Yes
IA	Interactive (default for GUI)	Yes	Yes	Yes

Use dispadmin (Dispatcher Admin) to get whole list of dispatchers available on your system.

```
global# dispadmin -l
CONFIGURED CLASSES
SYS      (System Class)        TS       (Time Sharing)
FSS      (Fair Share)          RT       (Real Time)
FX       (Fixed Priority)      IA       (Interactive)
```

5. The FSS (Fair Share Scheduler) in detail


DB Whole rooted Zone Solaris 10 Kernel 13888-07
DB Sparse Zone Solaris 10 Kernel 138888-07
App Whole rooted Zone, Solaris 9
App Whole rooted Zone, Solaris 8
1(6) Shares
6(1) Shares
2 Shares
2 Shares

With the FSS you can control the allocation of CPU resources among workloads. E.g. At nighttime you need more resources for your batch jobs for the dataware house, and during the business hours you need more resources for your OLTP system.
FSS guarantees a fair dispersion of CPU resources based on allocated shares. HINT: The FSS shares are relative. 2 shares does not mean 2 CPU's assigned to a zone. It means that if you have 10 shares on the overall system, you will get 20% of the CPU resources. When you don't assign any share, then the default is 1. Processes with zero shares always run at the lowest system priority.

5.1 Set FSS as the default scheduler for the global zone

To activate the FSS as your default scheduler, run the dispadmin command and reboot the machine. After the reboot you will see your new default scheduler in the /etc/dispadmin.conf file

```
global# dispadmin -d FSS; reboot
global# cat /etc/dispadmin.conf | grep DEFAULT
DEFAULT_SCHEDULER=FSS
```

5.2 Set FSS as the default scheduler for the local zone

To configure the FSS for your zone, run the zonecfg command, and apply the following settings. Remember that you can configure a different schedulers for the local zone, in case the FSS should not satisfy your needs. E.g. configuring the TS (Time Sharing) scheduler is also possible.

```
global# zonecfg -z tvdzone10o
zonecfg:tvdzone10o> set scheduling-class=FSS
zonecfg:tvdzone10o> set cpu-shares=6
zonecfg:tvdzone10o> commit
zonecfg:tvdzone10o> exit
global# zoneadm -z tvdzone10o reboot
```

After everything is set and done, we can start changing the FSS shares for the appropriate zones at runtime. E.g. we can change the number of FSS shares from 6 to 1 for the OLTP zone tvdzone10o.

```
global# prctl -n zone.cpu-shares -v 1 -r -i zone tvdzone10o
```

Now let's verify if we really have the correct number of shares.

```
global# prctl -n zone.cpu-shares -i zone tvdzone10o
zone: 10: tvdzone10o
NAME    PRIVILEGE       VALUE    FLAG   ACTION       RECIPIENT
zone.cpu-shares
        privileged          2       -   none         -
        system          65.5K     max   none         -
```

6. Creating a sparse zone that is ready for Oracle

There are several steps, which we got to do, to create a sparse zone that is ready for running an Oracle instance.

- Create the /zones filesystem on the global host. In case the /zones filesystem is ZFS and resides on a shared storage, then you can move the zones between the hosts
- Change the directory permissions to 700 for your Oracle zone /zones/tvdzone10o
- Create the /opt/tvdzone10o/local directory. This directory is needed, because Oracle wants to write some files into the /usr/local/bin directory, and this is per default read only in a sparse zone
- Create the CPU Pool
- Create the zone configuration for tvdzone10o (zonecfg)
- Install the zone (zoneadm)
- Boot the zone
- Login (zlogin) to the zone, and finish the installation

Generally speaking, with the tools zonecfg, zoneadm and zlogin most of the work can be done. In case you work with the resource management from Solaris, then you might also need pooladm, poolcfg, rcapadm, rcapstat, ipqosconf and some more tools.

7. Create the CPU Pool

Now let's create the CPU Pool that we would like to assign to our Zone. First of all, we need to activate the CPU Pool feature.

```
pooladm -e
```

Now, we create the file proc.txt with the following lines, and then run the poolcfg command.

```
create pset PsetDB (uint pset.min=1;uint pset.max=1)
create pool PoolDB
associate pool PoolDB ( pset PsetDB )

poolcfg -f proc.txt
```

8. Create the zone configuration, general settings

With the following commands, we create a blank (-b) configuration, and create everything by our own, without taking the defaults. The configuration is more or less self explanatory. I would like only to pick out the limitpriv settings.
With the proc_lock_memory, we allow Oracle to use the SGA_MAX feature, and with the dtrace_proc and dtrace_user settings, we are able to use the powerful dtrace Framework inside a zone. Notice that there are only a limited amount of dtrace probes available in a zone, not all of them.

```
global# zonecfg -z tvdzone10o
tvdzone10o: No such zone configured
Use 'create' to begin configuring a new zone.
tvdzone10o> create -b
tvdzone10o> set zonepath=/zones/tvdzone10o
tvdzone10o> set autoboot=true
tvdzone10o> set limitpriv=default,proc_lock_memory,
dtrace_proc,dtrace_user
tvdzone10o> set pool=PoolDB
tvdzone10o> set scheduling-class=FSS
tvdzone10o> set ip-type=shared
tvdzone10o> add net
tvdzone10o:net> set address=172.16.65.23
tvdzone10o:net> set physical=hme0
tvdzone10o:net> end
```

9. Create the zone configuration, sparse zone settings

With the following lines, we configure the directories, which are inherited from the global zone.

```
tvdzone10o> add inherit-pkg-dir
tvdzone10o:inherit-pkg-dir> set dir=/lib
tvdzone10o:inherit-pkg-dir> end
tvdzone10o> add inherit-pkg-dir
tvdzone10o:inherit-pkg-dir> set dir=/platform
tvdzone10o:inherit-pkg-dir> end
tvdzone10o> add inherit-pkg-dir
tvdzone10o:inherit-pkg-dir> set dir=/sbin
tvdzone10o:inherit-pkg-dir> end
tvdzone10o> add inherit-pkg-dir
tvdzone10o:inherit-pkg-dir> set dir=/usr
tvdzone10o:inherit-pkg-dir> end
tvdzone10o>
```

10. Create the zone configuration, file system settings

Now we assign the filesystems /u00 and /usr/local to the local zone.

```
tvdzone10o> add fs
tvdzone10o:fs> set dir=/u00
tvdzone10o:fs> set special=/zpool/tvdzone10ou00
tvdzone10o:fs> set type=lofs
tvdzone10o:fs> end
tvdzone10o> add fs
tvdzone10o:fs> set dir=/usr/local
tvdzone10o:fs> set special=/opt/tvdzone10o/local
tvdzone10o:fs> set type=lofs
tvdzone10o:fs> end
tvdzone10o>
```

11. Create the zone configuration, FSS share settings

With the following lines we configure the number or CPU shares, that we want to assign to the local zone.

```
tvdzone10o> add rctl
tvdzone10o:rctl> set name=zone.cpu-shares
tvdzone10o:rctl> add value (priv=privileged,limit=6,action=none)
tvdzone10o:rctl> end
tvdzone10o> verify
tvdzone10o> commit
tvdzone10o> exit
```

commit: Write changes to /etc/zones/tvdzone10o.xml file

12. Install and boot the zone

After the zone configuration is done, we are now ready to install and boot the zone.

```
global# zoneadm -z tvdzone10o install
Preparing to install zone <tvdzone10o>.
Creating list of files to copy from the global zone.
Copying <7676> files to the zone.
Initializing zone product registry.
Determining zone package initialization order.
Preparing to initialize <1119> packages on the zone.
Initialized <1119> packages on zone.
Zone <tvdzone10o> is initialized.
The file </zones/tvdzone10o/root/var/sadm/system/logs/install_log> contains
a log of the zone installation.

global# zoneadm -z tvdzone10o boot
```

13. Finishing the installation

```
global# zlogin -C tvdzone10o
...
...
Answer the installation questions: time zone, root passwd ... etc
```

That's it. We have created a capped Solaris 10 container, and we are ready to start the installation of the Oracle database.

14. Conclusion

From my point of view, running Oracle in a capped Solaris container is a good choice, when you want to use a robust, reliable and fast virtualization technology and if you want to minimize your Oracle license cost at the same time. Solaris containers are available for quite a long time, and many customers are already using the container technology in combination with Oracle. In case you need further details, or help to implement containers in your environment, please feel free to contact me.

William Sescu
Consultant

Trivadis GmbH
Lehrer-Wirth-Str. 4
D-81829 München

Phone +49-89-99 27 59 30
Fax +49-89-99 27 59 59
Mail: william.sescu@trivadis.com
Internet: http://www.trivadis.com

15. Sources

- Official website from SUN
    - http://www.sun.com
- Official website from Oracle
    - http://www.oracle.com