



An Oracle White Paper
June 2011

How to Move an Oracle[®] Solaris Container

| | |
|--|----|
| Introduction | 1 |
| Oracle Solaris Container Migration: Overview | 2 |
| Oracle Solaris Container Migration: Three Examples..... | 4 |
| Containers Residing on the Root Filesystem..... | 5 |
| Containers Residing on an Oracle Solaris ZFS Filesystem..... | 14 |
| Containers that Live in UFS/SVM Filesystems | 23 |
| For More information | 30 |

Introduction

This white paper instructs users, system administrators, and developers who have experience with Oracle® Solaris 10 on correct use of new features which allow an Oracle Solaris 10 container to be moved from one computer to another. The paper starts with a brief discussion of the need for such functionality and follows with three examples of the use of this functionality.

Users are guided step-by-step through the process of moving a container, with code examples and illustrations. After using this paper, a user should be able to move an Oracle Solaris container by:

- Creating a container
- Detaching the container
- Moving the container to a new computer
- Attaching the container to the new computer

For thoroughness, code examples are provided for three different scenarios:

- Each system has only one filesystem
- Each system has containers installed onto Oracle Solaris ZFS filesystems
- Each system has containers installed onto Oracle Solaris UFS filesystems using Oracle Solaris Volume Manager

Oracle Solaris Container Migration: Overview

Businesses are attempting to reduce the number of computers in their data centers without reducing their architectural flexibility or hobbling themselves with unnecessary administrative complexity. Several server virtualization technologies exist to help meet these goals, including a feature in the Oracle Solaris 10 OS called Oracle Solaris 10 Containers.

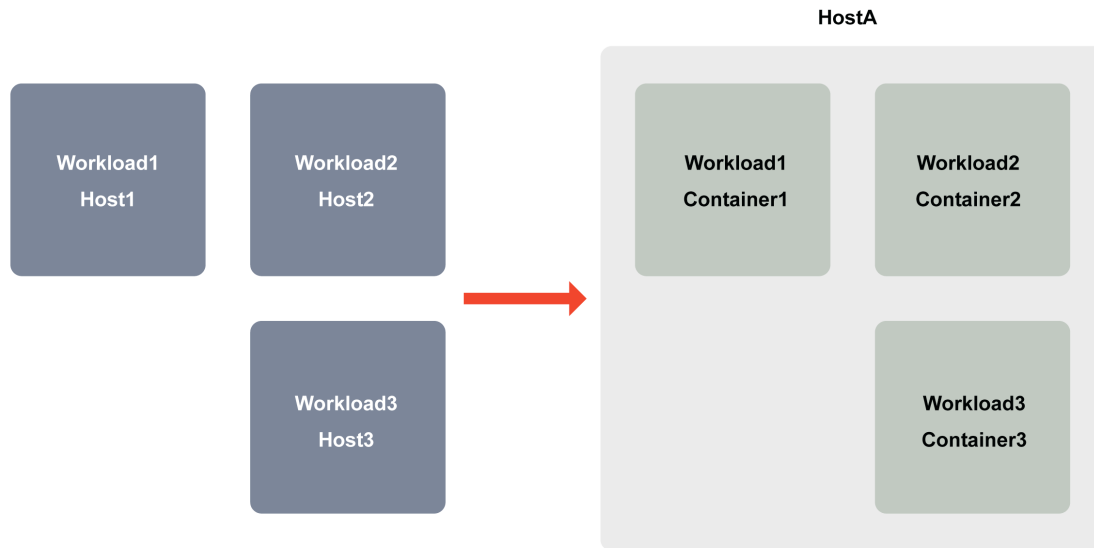


Figure 1—Consolidation of three workloads into one system

Architectural flexibility includes the ability to move a workload from one computer to another when necessary. Oracle Solaris 10 Containers include this capability. This document describes several scenarios that can be used to move a container from one computer to another.

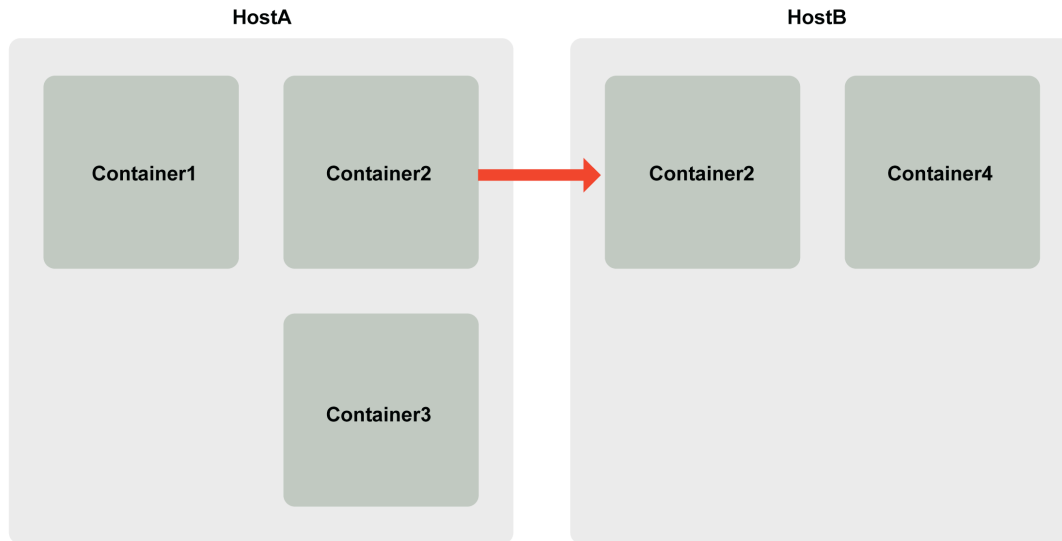


Figure 2—Movement of an Oracle Solaris container from one computer to another

The functionality of Oracle Solaris containers incorporates two categories of features: application isolation, referred to as Oracle Solaris Zones, and resource management which is made up of pools, projects, processor sets, the fair share scheduler, and kernel parameters.

Oracle Solaris Containers are very attractive to data centers attempting to consolidate many computers into fewer computers. However, conditions and requirements of a data center and its workloads often change and when that happens, the ability to move a container from one computer to another becomes very valuable. Without this ability, data center architects feel locked in to a configuration which may no longer be optimal.

As an example of server consolidation, a four-CPU Sun Fire V490 server running Oracle Solaris 10 may have four containers, each of which is running database software. In this example, each container has been configured with:

- Unlimited access to 8GB RAM
- Shared access to 8GB of swap space
- Shared use of internal SCSI bandwidth
- Access to shared files in /sbin, /usr, and other filesystems
- Access to private files in /etc, /var, and other filesystems, and 8GB of space allocated to a private filesystem
- Access to the network via a shared 1Gbps NIC
- An IP address
- 10 CPU shares

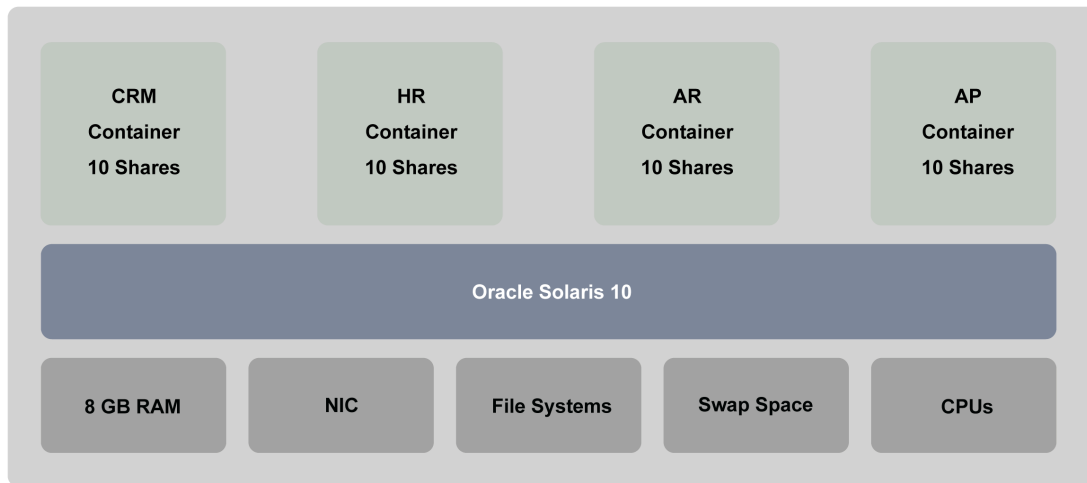


Figure 3—Example configuration

The Fair Share Scheduler (FSS) enables CPU resources to be allocated proportionally to applications. That is, each application gets assigned a number of shares which represent a portion of the total number of shares, which in turn represent the processing power of the whole system or of a resource pool. So the 10 CPU shares given to each of the containers above guarantee to one container's processes, in aggregate, a minimum of one-fourth of the processing capacity of the system, or approximately one CPU of processing power. This does not limit a container's processes to 25%—processing capacity does not go to waste if it is available.

As shown in Figure 3 one of the containers has a CRM database installed in it. In this example, use of the CRM database has been growing and occasionally experiences performance problems. The bottleneck could be insufficient hardware resources, e.g. storage bandwidth, CPU power, or memory. Performance analysis will determine the cause and, in some cases, the problem can be addressed by simply changing the container's configuration on the same system.

On the other hand, if the computer simply has insufficient compute capacity for the four workloads, the best solution may be to move the CRM container to another computer. This could be either a computer dedicated to this task or one with sufficient unused resources to provide acceptable performance for the CRM environment. The end result is the movement of the entire workload, including all configuration information, from the Sun Fire V490 server to another computer running Oracle Solaris 10. The remaining three containers would also benefit from reduced resource contention.

Oracle Solaris Container Migration: Three Examples

This document describes the process used to move a container from one system to another. Three different example methods are included for completeness. The difference between the methods is the type of filesystem which holds the container's private files:

- **Containers Residing on the Root Filesystem.** This is the simplest method. Its description focuses on the migration aspects.
- **Containers Residing on an Oracle Solaris ZFS Filesystem.** Oracle Solaris ZFS maximizes data robustness and enables the container designer to prevent one container from filling up a file system that another container is using. The combination of these two feature sets also reinforces the isolation which is a central factor of Oracle Solaris zones. The strengths that Oracle Solaris ZFS brings to this situation include better general robustness than existing filesystem-volume manager combinations, simpler command sequences, and endian-neutrality.
- **Containers Residing in a UFS Filesystem Built on an Oracle Solaris Volume Manager Metadevice.** This method is provided for those sites that have standardized on SVM.

Containers Residing on the Root Filesystem

This method focuses on the steps needed to move a container. It ignores considerations that are specific to the filesystem type.

Create a Sparse Root Container

The first step in the process is creating a sparse-root container. Complete details of container creation are discussed at <http://www.oracle.com/technetwork/documentation/index.html>, for this example you only need to consider these factors:

- The container's name will be 'twilight'
- The container will be booted manually, not automatically at system boot time
- The Global zone directory which will house the container will be `/zones/roots/twilight`
- The container will use network port `hme0` and will have one IP address: `192.168.0.102`

The first command in this step is 'zonecfg' which merely records your configuration directives. The example below shows how to enter these directives interactively. Is it also possible to enter them into a file, and use the `-f` option to `zonecfg(1M)`.

Note that you will probably use a different network device name (e.g. `hme0`) and IP address.

1. Create a sparse-root container:

```

hostA-global# zonecfg -z twilight
zonecfg:twilight> create
zonecfg:twilight> set zonepath=/zones/roots/twilight
zonecfg:twilight> add net
zonecfg:twilight:net> set physical=hme0
zonecfg:twilight:net> set address=192.168.0.102
zonecfg:twilight:net> end
zonecfg:twilight> exit

```

Now that Oracle Solaris knows the characteristics this container should have, installing the necessary files is simple, but can take 5 to 30 minutes, depending almost entirely on the speed of the system's disk(s) which contain this filesystem.

2. Install the container:

```

hostA-global# zoneadm -z twilight install

```

Now place the container's system identification information in the right location. This is the most efficient way to quickly get the container ready to run.

3. To create the sysidcfg file:

```

hostA-global# cat > /zones/roots/twilight/root/etc/sysidcfg
system_locale=C
terminal=dtterm
network_interface=primary {
hostname=twilight
}
timeserver=localhost
security_policy=NONE
name_service=NONE
timezone=US/Eastern
root_password=""
^D

```

The last piece of information that the container needs can be supplied simply by creating an empty file with the following name.

```

hostA-global# touch /zones/roots/twilight/root/etc/.NFS4inst_state.domain

```

4. To create the file:

Boot the Container and Verify it Works

At this point, a bootable container resides on a host computer. For future reference, the configuration can be diagrammed like this:

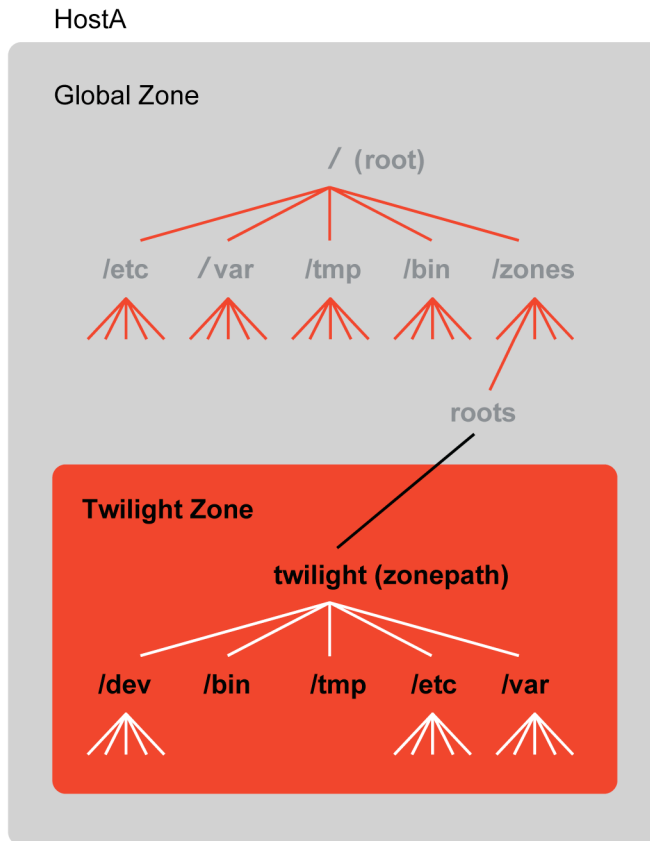


Figure 4—Container installed in HostA, but not running

Booting a container is both simple and quick.

1. To boot the container:

```
hostA-global# zoneadm -z twilight boot
```

HostA

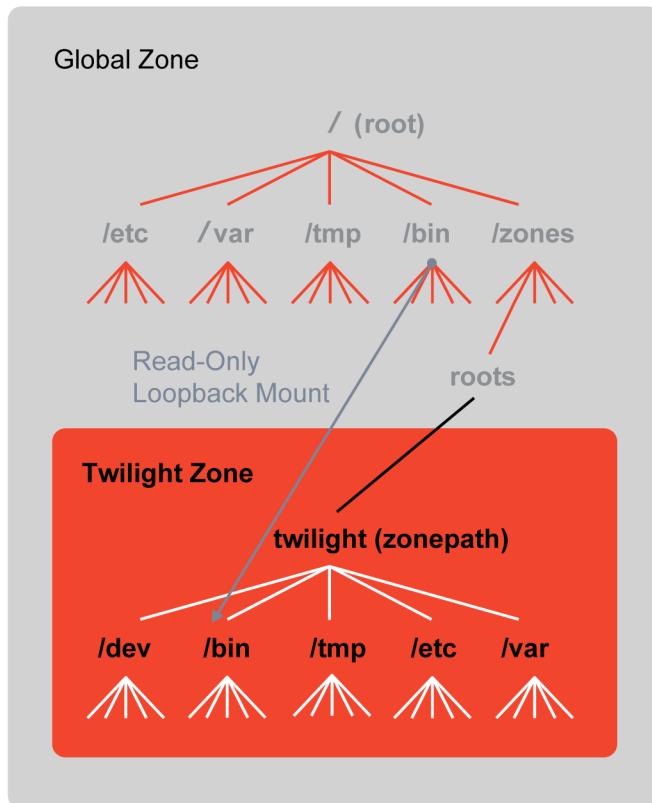


Figure 5—Booting a container creates loopback mounts

- To log into the console, use `zlogin`:

```
hostA-global# zlogin -C twilight
```

The `sysidcfg` file above specified that the root account does not have a password, so just press the **ENTER** key at the password prompt.

- Enter the `'hostname'` and `'zonename'` commands to see that the hostname is, by default, the same as the zonename:

```
twilight# hostname
twilight
twilight# zonename
twilight
```

- To end the `zlogin` session, enter these commands:

```
exit  
~.
```

Halt the Container

Before moving a container you must first halt it and all of its applications.

1. To halt the container, use the `zlogin` command:

Move the Container

The first step to move a halted container is detaching it from its original system. This step stores information about the container, including package and patch information. The information is stored in the same directory as the container's 'root' directory to simplify the process of gathering and moving the container's files.

1. To detach the container, use the `zoneadm` command:

```
hostA-global# zoneadm -z twilight detach
```

Once the container is detached, there are several ways to move it from one system to the other. This example will use `pax(1)` and `scp(1)`. The next steps create a pax archive file which contains a copy of all of the files in the container's root filesystem, moves the archive to the destination system, and then unpacks the archive.

2. To create a pax archive, use the `pax` command:

```
hostA-global# cd /zones/roots/twilight  
hostA-global# pax -w@f /tmp/twilight.pax -p e *
```

This diagram shows that the original computer now has a container and an archive of the container's private files:

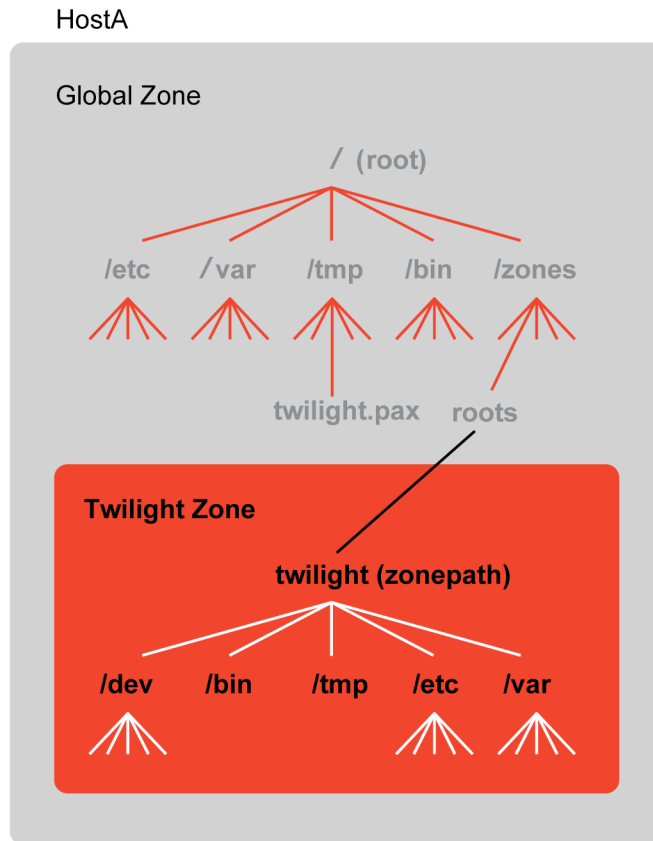


Figure 6—HostA after the container has been detached and archived

- To copy the archive to the new system, use the scp command:

```
hostA-global# scp /tmp/twilight.pax root@hostB:/tmp/twilight.pax
```

The new container will have a new zonename, 'dusk' but will retain the original hostname and IP address. Because the new system has the same hardware configuration as the original system, the new container will also use the hme0 network device.

Now that the archive is on the new system, the rest of the work will happen on the new system as well. Note that the command-line prompt will change to indicate this. The change of zonename is not a requirement of the technology, but demonstrates how you might use it to create a gold master container and copy it out to different systems.

- To unpack the archive, create a directory and use the pax command:

```

hostB-global# mkdir -m 700 -p /zones/roots/dusk
hostB-global# cd /zones/roots/dusk
hostB-global# pax -r@f /tmp/twilight.pax -p e

```

At this point, the files have been moved, but you still must attach the container to the system.

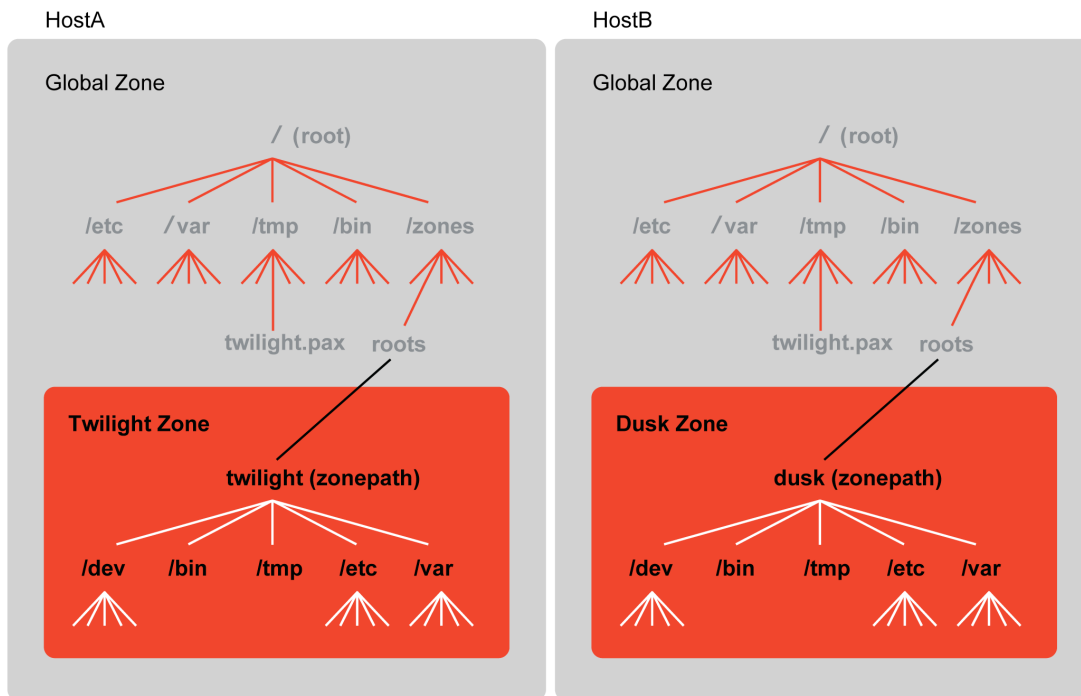


Figure 7—After archive has been copied to HostB and unpacked

Attach the Container and Verify that it Works

The final steps in moving a container are creating the container configuration and attaching the newly unpacked container files. Creation of the new container configuration is much simpler than the original container creation, because the configuration of the original container was stored in the archive.

Note that if the hardware configuration of the destination system is different, the configuration specification below must also implement those changes. Also, if the primary network interface of the new system is on a different subnet than the primary network interface of the old system, modifications will need to be made to some combination of the container's IP address, the set of IP addresses which exist on the new system, and the routing table of the new system. Such changes are beyond the scope of this document.

1. To create the new container configuration, use the zonecfg command:

```

hostB-global# zonecfg -z dusk
zonecfg:dusk> create -a /zones/roots/dusk
zonecfg:dusk> exit

```

Note that the '-a' option to the 'create' sub-command of zonecfg means “attach the container files that exist at this directory as this new container.”

Now that the new container has been configured, the container files that were copied over from the other system can be attached.

- To attach the container to the new system, use the zoneadm command:

```

hostB-global# zoneadm -z dusk attach

```

The process of attachment includes a check to ensure that the container being moved is compatible with the new system, including patch revisions.

At this point both containers exist and are identical except for their zonenames and the fact that the container on the first system is still in a detached state.

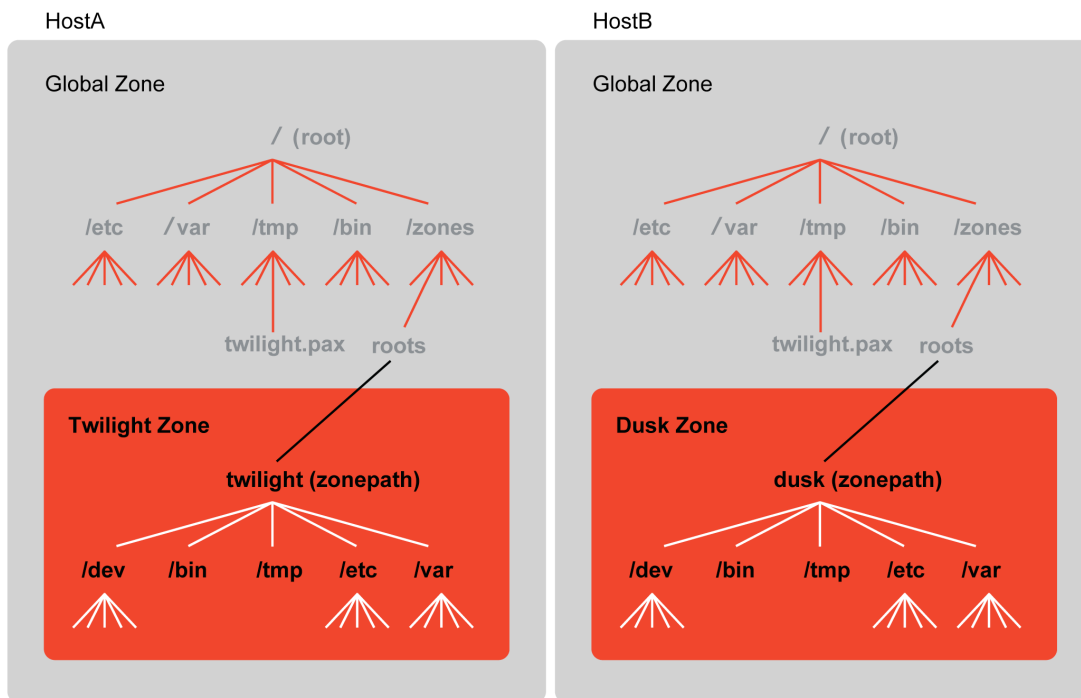


Figure 8—After container 'dusk' has been attached to HostB

The container is ready. It's time to boot it and use it!

- To boot the container, use the zoneadm command:

```
hostB-global# zoneadm -z dusk boot
hostB-global# zlogin -C dusk
```

Although the zonename has changed, note that the hostname and IP address have not.

4. Check the zonename, Hostname, and IP address of the new container:

```
twilight# zonename
dusk
twilight# hostname
twilight
twilight# ifconfig -a
...
hme0:1: flags=1000823<UP,BROADCAST,NOTRAILERS,MULTICAST,IPv4> mtu 1500 index
2 inet 192.168.0.102 netmask ffffffff broadcast 192.168.0.255
```

Note however, that these three zone attributes, as well as many others, are configurable. For example, the new container could have had the same zonename as the original, and you can change the hostname and IP address whenever you want.

This completes the detachment and reattachment of a zone, effectively moving it from one server to another.

Containers Residing on an Oracle Solaris ZFS Filesystem

The preceding section demonstrated the simplest possible scenario of container migration. One of the weaknesses of that method is that one container could fill up the computer's root filesystem, causing a denial-of-service attack. To prevent this, each container could be installed into its own filesystem. In some situation, the maximum number of filesystems per disk would limit the number of containers on a system. However, both Oracle Solaris ZFS and Oracle Solaris UFS/SVM enable the creation of large number of filesystems per disk. Doing so allows large numbers of containers per system and also reinforces container isolation. The next two sections will demonstrate this ability.

This section demonstrates steps which could be used to install a container in an Oracle Solaris ZFS filesystem. The filesystem is of limited size, preventing this container from limiting the filesystem space available to other containers.

This section uses the same basic method to create, detach and reattach the container as the previous example. For brevity, the commands to perform those steps are replicated below, but not explained.

Create an Oracle Solaris ZFS Filesystem for the Container

Oracle Solaris ZFS filesystems are created using the disk space assigned to an Oracle Solaris ZFS pool. Unlike the mapping of a single UFS filesystem to an Oracle Solaris Volume Manager, multiple Oracle Solaris ZFS filesystems can be assigned to one Oracle Solaris ZFS pool.

To create an Oracle Solaris ZFS pool, simply specify the name of the pool and the disk drives that it will use. Note that although this example specifies whole disk drives, individual disk partitions could be used instead. Also, this example does not employ a data redundancy method such as disk mirroring. See the `zpool(1M)` man page for more information.

1. To create a new Oracle Solaris ZFS pool, use the `zpool` command:

```
hostA-global# zpool create zone_roots c1t0d0 c1t0d1 c1t0d2 c1t0d3
```

Now that the pool exists, you can tell Oracle Solaris ZFS to create a filesystem in it. Note that the name of the filesystem is `zone_roots/zfszone1` and that these commands do not specify a leading '/'. However, by default the filesystem will appear at `/zone_roots/zfszone1`. See the man page for `zfs(1M)` for a complete description of this.

2. To create a new Oracle Solaris ZFS filesystem, use the `zfs` command:

```
hostA-global# zfs create zone_roots/zfszone1
hostA-global# chmod 700 /zone_roots/zfszone1
```



Figure 9—An Oracle Solaris ZFS Pool and its Filesystems

Assign a capacity of 500MB for the container. Note that this capacity can be increased or decreased later very easily.

3. To set the quota on the filesystem, use the `zfs` command:

```
hostA-global# zfs set quota=500m zone_roots/zfszone1
```

At this point you may have noticed one advantage of Oracle Solaris ZFS over common filesystem-volume management systems: management simplicity. For example, it will take approximately three times as many commands in UFS/SVM or VxFS/VxVM environments to achieve the results achieved by the four Oracle Solaris ZFS commands shown above.

Create a Sparse-root Container and Verify that it Works

For your convenience all of the commands needed to create a new container on the first system are listed below. For a full explanation see the section above entitled *Containers Residing on the Root Filesystem*.

```
hostA-global# zonecfg -z zfszone1
zonecfg:zfszone1> create
zonecfg:zfszone1> set zonpath=/zone_roots/zfszone1
zonecfg:zfszone1> add net
zonecfg:zfszone1:net> set physical=hme0
zonecfg:zfszone1:net> set address=192.168.0.102
zonecfg:zfszone1:net> end
zonecfg:zfszone1> exit
hostA-global# zoneadm -z zfszone1 install
hostA-global# cat > /zone_roots/zfszone1/root/etc/sysidcfg
system_locale=C
terminal=dtterm
network_interface=primary {
hostname=zfszone1
}
timeserver=localhost
security_policy=NONE
name_service=NONE
timezone=US/Eastern
root_password=""
^D
hostA-global# touch /zone_roots/zfszone1/root/etc/.NFS4inst_state.domain
hostA-global# zoneadm -z zfszone1 boot
hostA-global# zlogin -C zfszone1
```

The Oracle Solaris ZFS filesystem provides a number of commands that let you observe filesystem usage and state. Although the following two commands are not necessary to migrate a container, they highlight how little disk space the container is actually using.

1. To list the current Oracle Solaris ZFS filesystems, use the `zfs list` command:

```
hostA-global# zfs list
NAME                                USED    AVAIL    REFER    MOUNTPOINT
zone_roots                          73.3M   24.9G   9.50K    /zone_roots
zone_roots/zfszone1                 72.9M   427M    72.9M    /zone_roots/zfszone1
```

Note that although the pool has 25GB of disk space the filesystem only has 500MB available from the pool, and has only used 73MB for the container.

2. To look at three 3-second interval I/O statistics on the active Oracle Solaris ZFS filesystems, use the `zpool iostat` command:

```
hostA-global# zpool iostat 3 3

           capacity           operations           bandwidth
pool      used  avail    read  write    read  write
-----
zone_roots 73.3M 25.1G     0     1     492  70.2K
zone_roots 73.3M 25.1G     0     1         0     0
zone_roots 73.3M 25.1G     0     1         0     0
```

More information on using Oracle Solaris ZFS commands can be found in the Oracle Solaris ZFS Administration Guide.

Now the container must be halted in order for it to be moved to the other system.

3. To halt the container, use the `zlogin` command:

```
hostA-global# zlogin zfszone1 shutdown -y -i 0
```

Detach the Container, Create an Archive of it, and Move it to the Destination System

The container is now ready to be detached and moved.

1. To detach the container, use the `zoneadm` command:

```
hostA-global# zoneadm -z zfszone1 detach
```

This method will use the backup feature of Oracle Solaris ZFS. The backup feature only operates on a snapshot, to improve data integrity. So first create a snapshot of the filesystem:

2. To create an Oracle Solaris ZFS snapshot, use the `zfs` command:

```
hostA-global# zfs snapshot zone_roots/zfszone1@Snap1
```

HostA

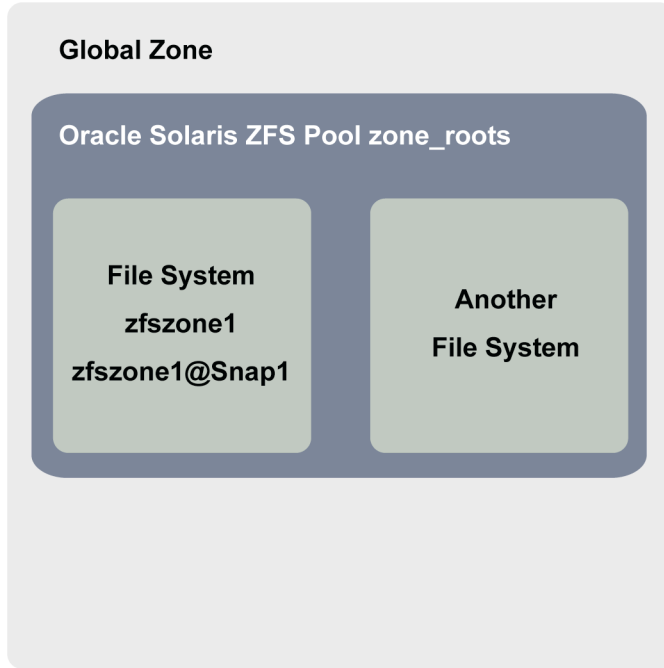


Figure 10—An Oracle Solaris ZFS Snapshot

The Oracle Solaris ZFS 'send' sub-command packages all of the files and directories in an Oracle Solaris ZFS filesystem and archives them. For this example, simply put the archive into a file.

3. To back up the Oracle Solaris ZFS snapshot, use the zfs command:

```
hostA-global# zfs send zone_roots/zfszone1@Snap1 > /tmp/zfszone1.Backup1
```

HostA

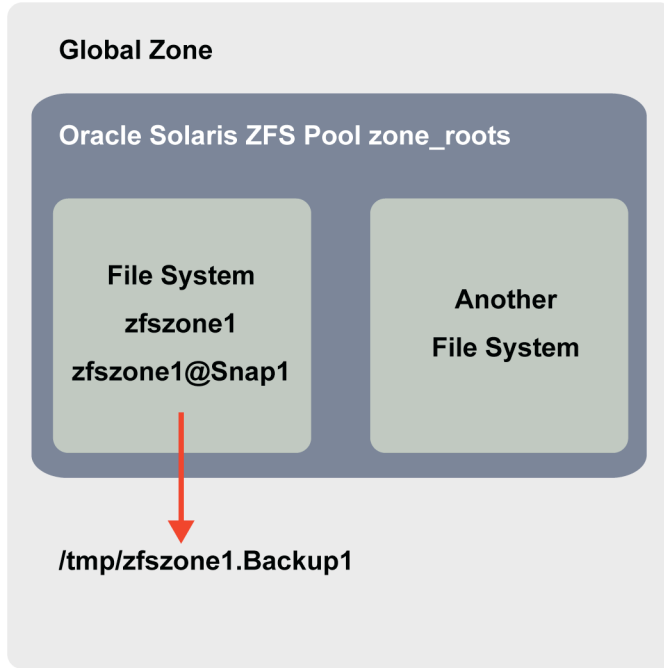


Figure 11—An Oracle Solaris ZFS Archive

Move the archived copy of the container to the new system. Here is an example of the syntax for the `scp` command on properly configured systems:

4. To copy the archive, use the `scp` command:

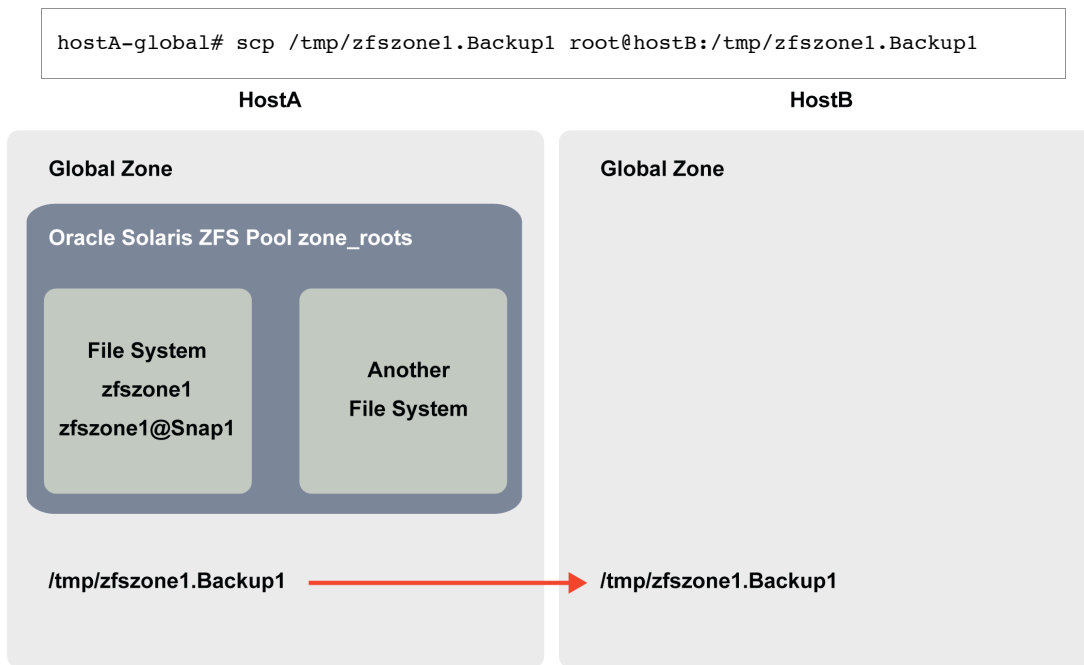


Figure 12—Copying an Oracle Solaris ZFS Archive

Create a New Pool on the New System and Restore the Archive into It

Now that the archive has been moved to the new system, all the rest of the work will happen on the new system. Note that the command-line prompt will change to indicate this.

The Oracle Solaris ZFS `receive` sub-command will be used to unpack the archive. Although it automatically creates a filesystem, it needs a pool into which the filesystem will be restored.

1. To create a new Oracle Solaris ZFS pool on the other system, use the `zpool` command:

```
hostB-global# zpool create zone_roots c1t0d0 c1t0d1 c1t0d2 c1t0d3
```

Now the filesystem can be restored into the pool. The only parameter that must be specified is the name of the new filesystem.

2. To restore the Oracle Solaris ZFS filesystem on the other system, use the `zfs` command:

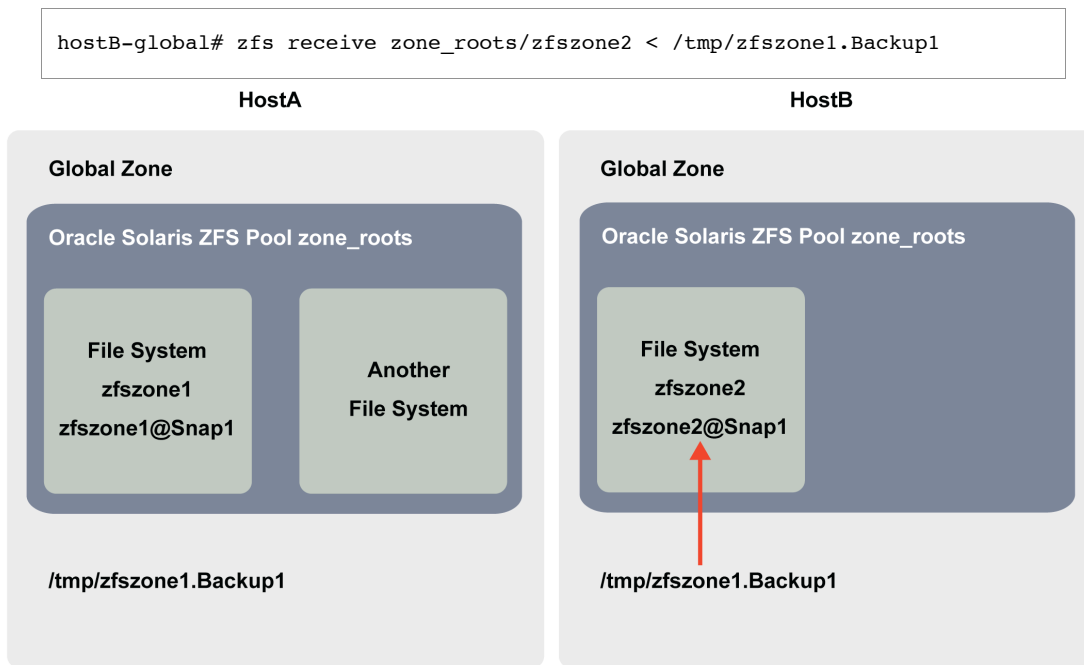


Figure 13—Unpacking an Oracle Solaris ZFS Archive

- To list the current Oracle Solaris ZFS state, use the `zfs` command:

```
hostB-global# zfs list
NAME                                USED    AVAIL    REFER    MOUNTPOINT
zone_roots                          73.3M   24.9G    10K      /zone_roots
zone_roots/zfszone2                  73.0M   24.9G    73.0M    /zone_roots/zfszone2
zone_roots/zfszone2@Snap1            0       -        73.0M    -
```

Note that the steps to backup and restore a filesystem do not carry the original filesystem's quota setting of 500MB. If a quota is needed on the new filesystem, it must be set after restoring the filesystem. Also note that the snapshot created in order to backup the filesystem gets restored along with the filesystem. The snapshot can be deleted to save disk space and simplify the Oracle Solaris ZFS configuration.

Attach the Container and Verify that it Works

Finally, everything is in place to correctly attach the container to the new system.

- To create a new configuration for the container, use the `zonecfg` command:

```

hostB-global# zonecfg -z zfszone2
zonecfg:zfszone2> create -a /zone_roots/zfszone2
zonecfg:zfszone2> exit

```

2. To attach and boot the container, use the zoneadm command:

```

hostB-global# zoneadm -z zfszone2 attach
hostB-global# zoneadm -z zfszone2 boot

```

3. Now check the container's zonename, hostname and IP address:

```

twilight# zonename
zfszone2
twilight# hostname
zfszone1
twilight# ifconfig -a
...
hme0:1: flags=1000823<UP,BROADCAST,NOTRAILERS,MULTICAST,IPv4> mtu 1500 index
2 inet 192.168.0.102 netmask ffffffff broadcast 192.168.0.255

```

Although the zonename has changed, note that the hostname and IP address have not. This section has demonstrated the detachment and reattachment of a zone, effectively moving it from one server to another. A few of the features of Oracle Solaris ZFS that illustrate the ease of administration of Oracle Solaris ZFS filesystems were also shown.

Containers that Live in UFS/SVM Filesystems

This method achieves the same goals as the previous section which used Oracle Solaris ZFS. Instead of Oracle Solaris ZFS, it uses the UNIX File System (UFS) and the Oracle Solaris Volume Manager (SVM). To achieve a large number of filesystems per disk, SVM soft partitions are used.

Although it is not necessary to use SVM or soft partitions to install containers in a UFS filesystem, it is the simplest method to implement many containers with UFS and still ensure that one container does not starve other containers of filesystem space. Specifically, soft partitions allow dozens of filesystems on one disk—without them, you must limit yourself to a handful of containers per disk or install multiple zones per disk slice. The latter solution allows one container to use all of the free space of the filesystem, potentially preventing the other containers from operating correctly.

As in the Oracle Solaris ZFS example, this section uses the same basic method to create, detach, and reattach the container. So again, the commands to perform those steps are replicated below, but not explained.

Create a SVM Soft Partition and a UFS Filesystem on the Soft Partition

This example uses two disks to create a 5GB mirrored metadevice, and from that creates a 200MB soft partition for the container to use.

Before configuring SVM for your own systems, you must understand the rules which specify the number and placement of the metadevice state database replicas. They are described in the `metadb(1M)` man page. This example uses two disks for the data and replicas. In this case, the rules dictate that two replicas must be created on each disk. In addition, one partition of 5GB is defined on each disk as a mirror into which all of the containers will be installed. The example uses the devices `c1t4d0s0`, `c1t4d0s1`, `c2t12d0s0`, and `c2t12d0s1` as the SVM database partitions, and devices `c1t4d0s3` and `c2t12d0s3` as the mirrors. A description of the commands to create these disk partitions is beyond the scope of this document.

The example below assumes the following disk layout:

TABLE 1. DISK LAYOUT

| | | |
|------------------------|------|----------------|
| <code>c1t4d0s0</code> | 20MB | metadata DB |
| <code>c1t4d0s1</code> | 20MB | metadata DB |
| <code>c1t4d0s3</code> | 5GB | data partition |
| <code>c2t12d0s0</code> | 20MB | metadata DB |
| <code>c2t12d0s1</code> | 20MB | metadata DB |
| <code>c2t12d0s3</code> | 5GB | data partition |

After creating the disk partitions, create the SVM database and the replicas of it.

```
hostA-global# metadb -a -f c1t4d0s0 c1t4d0s1 c2t12d0s0 c2t12d0s1
```

1. To create the SVM database and replicas, use the `metadb` command:

Next create two 'metadisks'—virtual devices consisting of a disk partition each.

2. To create the metadisks, use the `metainit` command:

```
hostA-global# metainit d11 1 1 c1t4d0s3
hostA-global# metainit d12 1 1 c2t12d0s3
```

Now create a third metadisk—another virtual device—in order to create a mirrored virtual device. This command also specifies one of the two mirrors.

3. To create the first part of the mirror, use the `metainit` command:


```
hostA-global# metainit d10 -m d11
```

The second virtual device can now be added as a mirror of the first metadisk.

- To add the second metadisk to the mirror, use the metattach command:

```
hostA-global# metattach d10 d12
```

HostA

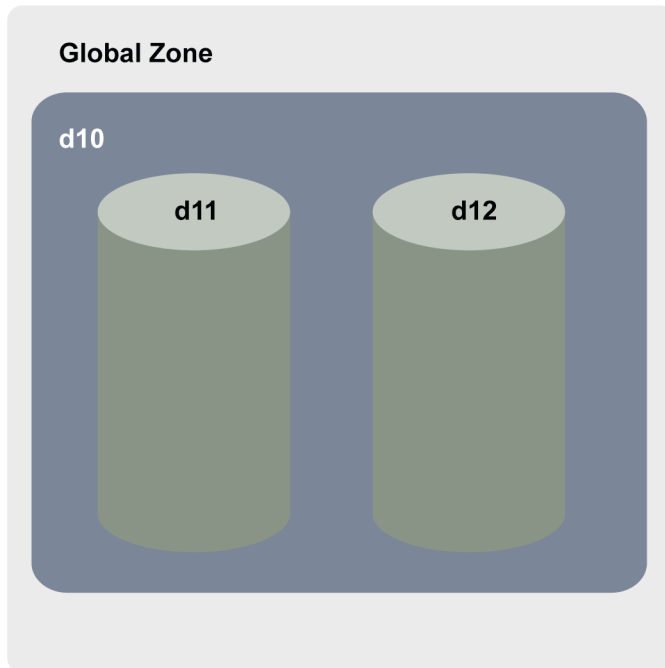


Figure 14—Oracle Solaris Volume Manager metadevices

A *soft partition* is an SVM feature which allows the creation of multiple virtual partitions in one metadisk. Creating one requires the `-p` option to `metainit`.

- To create a new soft partition, use the `metainit` command:

```
hostA-global# metainit d100 -p d10 200M
```

Once the volume exists and a capacity has been applied to it, a UFS filesystem can be created on it.

- To create the new UFS filesystem, use the `mkdir`, `newfs`, `mount`, and `chmod` commands:

```

hostA-global# mkdir -p /zones/roots/ufszone1
hostA-global# newfs /dev/md/dsk/d100
newfs: construct a new file system /dev/md/rdisk/d100: (y/n)? y
...
hostA-global# mount /dev/md/dsk/d100 /zones/roots/ufszone1
hostA-global# chmod 700 /zones/roots/ufszone1

```

If you want Oracle Solaris to mount that device the next time that the system boots, you must also add an appropriate entry in `/etc/vfstab`. For this example, the entry would look like this:

```

/dev/md/dsk/d100 /dev/md/rdisk/d100 /zones/roots/ufszone1 ufs 1 yes -

```

HostA

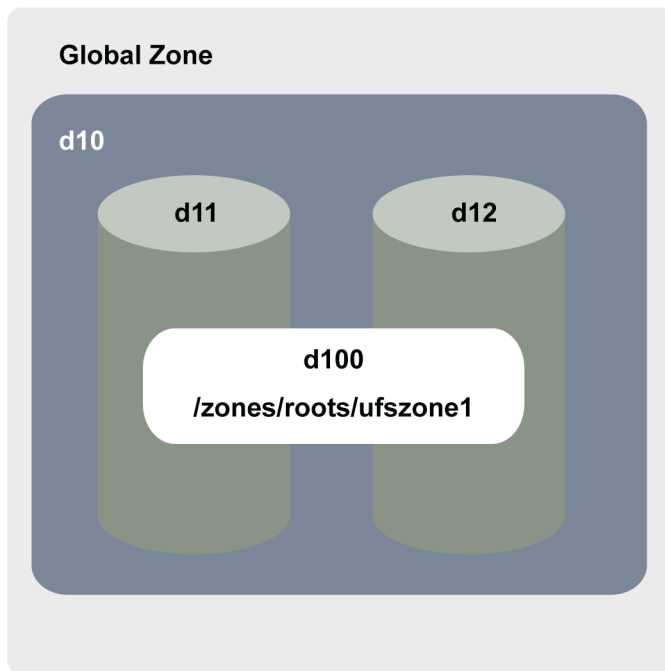


Figure 15—Oracle Solaris Volume Manager Soft Partition

Create a Sparse-Root Container, and Verify that it Works

For your convenience all the commands needed to create a new container on the first system are listed below. For a full explanation see the section above.

```

hostA-global# zonecfg -z ufszone1
zonecfg:ufszone1> create
zonecfg:ufszone1> set zonepath=/zones/roots/ufszone1

```

```

zonecfg:ufszonel> add net
zonecfg:ufszonel> set physical=hme0
zonecfg:ufszonel> set address=192.168.0.102
zonecfg:ufszonel> end
zonecfg:ufszonel> exit
hostA-global# zoneadm -z ufszone1 install
hostA-global# cat > /zones/roots/ufszonel/root/etc/sysidcfg
system_locale=C
terminal=dtterm
network_interface=primary {
hostname=ufszonel
}
timeserver=localhost
security_policy=NONE
name_service=NONE
timezone=US/Eastern
root_password=""
^D
hostA-global# touch /zones/roots/ufszonel/root/etc/.NFS4inst_state.domain
hostA-global# zoneadm -z ufszone1 boot
hostA-global# zlogin -C ufszone1

```

The sysidcfg file above specified that the root account does not have a password, so just press the **ENTER** key at the password prompt. Enter the 'hostname' and 'zonename' commands to see that the hostname is, by default, the same as the zonename.

```

ufszonel# hostname
ufszonel
ufszonel# zonename
ufszonel

```

1. Check the zonename and hostname of the new container:
2. To end the zlogin session, enter these commands:

```

exit
~.

```

3. To halt the container so it can be moved, use the zlogin command:

```
hostA-global# zlogin ufszone1 shutdown -y -i 0
```

Detach the Container and Move it

Now the container is halted and ready to be moved to the other system. First, detach the container from the system.

1. To detach the container from the system, use the zoneadm command:

```
hostA-global# zoneadm -z ufszone1 detach
```

The Oracle Solaris pax(1) command is a hardware-independent archival tool. The syntax shown below creates an archive file named /tmp/ufszone1.pax.

2. To package the container files into a single archive, first cd into the zonepath and then use the pax command:

```
hostA-global# cd /zones/roots/ufszone1
hostA-global# pax -w@f /tmp/ufszone1.pax -p e *
```

As in earlier examples, move the archive file with any file transfer method.

3. To move the archive securely, use the scp command:

```
hostA-global# scp /tmp/ufszone1.pax root@hostB:/tmp/ufszone1.pax
```

Create an SVM Soft Partition and UFS Filesystem on the New System and Restore the Archive into it

On the new system, repeat the creation of a filesystem using UFS and SVM as illustrated above, substituting the word 'ufszone2' for the word 'ufszone1'. Make sure you create the /zones/roots/ufszone2 filesystem.

When the filesystem is ready, unpack the container-archive on the new system.

1. To unpack the archive on the new system, cd to the zonepath and use the pax command:

```
hostB-global# cd /zones/roots/ufszone2
hostB-global# pax -r@f /tmp/ufszone1.pax -p e
```

Attach the Container and Verify that it Works

Now that the filesystem is created and the container files are unpacked into it, the container can be configured and attached to the new system.

1. To configure the container, use the zonecfg command:

```

hostB-global# zonecfg -z ufszone2
zonecfg:ufszone2> create -a /zones/roots/ufszone2
zonecfg:ufszone2> exit

```

2. To attach and boot the container, use the zoneadm command:

```

hostB-global# zoneadm -z ufszone2 attach
hostB-global# zoneadm -z ufszone2 boot

```

The container is now up and running and, as before, you can log in to it and check things like the hostname, zonename, and IP address.

3. To log in to the container on its console, use the zlogin -C command:

```

hostB-global# zlogin -C ufszone2
ufszone1# hostname
ufszone1
ufszone1# zonename
ufszone2
ufszone1# ifconfig -a
...
hme0:1: flags=1000823<UP,BROADCAST,NOTRAILERS,MULTICAST,IPv4> mtu 1500 index
2 inet 192.168.0.102 netmask ffffffff broadcast 192.168.0.255

```

Note that although the zonename has changed, the hostname and IP address have not.

This section has demonstrated the use of UFS and SVM with detachment and reattachment of a zone, effectively moving it from one server to another.

For More information

While this Oracle Solaris white paper provides the basic steps needed to move Oracle Solaris containers from system to system using several filesystems, including the innovative Oracle Solaris ZFS filesystem, it is only an introduction to the possibilities for improving your Oracle Solaris container implementations—more sophisticated configurations are possible. For more information regarding Oracle Solaris containers, the Oracle Solaris ZFS filesystem and other topics touched upon in this paper, visit www.oracle.com/us/products/servers-storage/solaris/index.html.



How to Move an Oracle Solaris Container
May 2010
Author: Jeff Victor

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com



Oracle is committed to developing practices and products that help protect the environment

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. UNIX is a registered trademark licensed through X/Open Company, Ltd. 0310

SOFTWARE. HARDWARE. COMPLETE.