# Creating and Using Oracle® Solaris Kernel Zones

**Part No: E54751**
November 2016

Creating and Using Oracle Solaris Kernel Zones

**Part No: E54751**

**Documentation Accessibility**

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

**Access to Oracle Support**

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

**Référence: E54751**

**Accessibilité de la documentation**

Pour plus d’informations sur l’engagement d’Oracle pour l’accessibilité à la documentation, visitez le site Web Oracle Accessibility Program, à l’adresse `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

**Accès aux services de support Oracle**

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d’informations, visitez le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` ou le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` si vous êtes malentendant.

# Contents

# Using This Documentation

- **Overview** – Describes how to plan, configure, install, and administer Oracle Solaris Zones
- **Audience** – Technicians, system administrators, and authorized service providers
- **Required knowledge** – Experience administering Oracle Solaris environments. Experience with virtualized environments is a plus.

## Product Documentation Library

Documentation and resources for this product and related products are available at `http://www.oracle.com/pls/topic/lookup?ctx=E53394`.

## Feedback

Provide feedback about this documentation at `http://www.oracle.com/goto/docfeedback`.

♦ ♦ ♦ CHAPTER 1

# Planning and Configuring Oracle Solaris Kernel Zones

This chapter discusses how to plan and configure Oracle Solaris Kernel Zones, also known as `solaris-kz` branded zones. It provides procedures for verifying hardware support, zone configuration, and applying zone resources specific to kernel zones.

This chapter covers the following planning and configuration topics:

## About Oracle Solaris Kernel Zones

An Oracle Solaris Kernel Zone, also called a `solaris-kz` branded zone, uses the branded zones framework to run a zone with a separate kernel and operating system (OS) installation from the global zone. The separate kernel and OS installation provide for greater independence and enhanced security of operating system instances and applications.

The administrative and structural content of a kernel zone is entirely independent from that of the global zone. For example, a kernel zone does not share system packaging with the global zone, or kernel zone host. Package updates on the kernel zone host are not linked images and do not affect kernel zones. Similarly, packaging commands such as `pkg update` are fully functional from inside of a kernel zone. See Chapter 3, “Installing, Removing, and Updating Software Packages” in *Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3* for additional information on packaging commands.

System processes are handled in the kernel zone's separate process ID table and are not shared with the global zone. Resource management in kernel zones is also different. Resource controls such as `max-processes` are not available when configuring a kernel zone.

The `zoneadm rename` command is not supported for kernel zones in the installed state. You can only change the name of a kernel zone by using the `zonecfg` command. The kernel zone must be in the configured or the unavailable state.

Use the existing `zlogin`, `zonecfg`, and `zoneadm` commands to manage and to administer kernel zones on the global zone.

For more information about the branded zones framework, see the `brands(5)` man page.

See Chapter 1, "Oracle Solaris Zones Introduction" in *Introduction to Oracle Solaris Zones* for additional overview of kernel zones concepts.

**Caution -** An Oracle Solaris x86 system should not run Oracle VM VirtualBox and Oracle Solaris Kernel Zones at the same time.

## Kernel Zones and General Zones Concepts

This manual assumes that you are familiar with the following resource management and zones concepts:

- Resource controls that determine how applications use available system resources
- Commands used to configure, install, and administer zones, primarily `zonecfg`, `zoneadm`, and `zlogin`
- `zonecfg` resources and property types
- Global zones and non-global zones
- The whole-root non-global zone model
- Authorizations granted through the `zonecfg` utility
- Global administrator and zone administrator
- The zone state model
- Zone isolation characteristics
- Network concepts and configuration
- Zone exclusive-IP type

See *Introduction to Oracle Solaris Zones* and *Creating and Using Oracle Solaris Zones* for more information about these concepts.

## Hardware and Software Requirements for Oracle Solaris Kernel Zones

To use the latest features in kernel zones, such as live migration, you must be running at least Oracle Solaris 11.3 on your host operating system.

The physical machine must meet the following requirements.

**SPARC based systems:**

- A SPARC T4 series server with at least System Firmware 8.8.
- A SPARC T5, SPARC M5, or SPARC M6 series server with at least System Firmware 9.5.
- A SPARC T7 or SPARC M7 series server. All firmware versions are supported.
- To run kernel zones, a Fujitsu M10 or SPARC M10 server with XCP Firmware 2230 or newer and Oracle Solaris 11.3 or newer.

  To use the kernel zones live migration feature on Fujitsu M10 systems, follow the firmware requirements in *Fujitsu M10 Systems Product Notes*.

See the web page Firmware Downloads and Release History for Oracle Systems for information about downloading the latest system firmware.

**x86 based systems:**

- Intel-based systems must have Nehalem or later processors
- AMD-based systems must have Barcelona or later processors
- BIOS must enable the following features:
  - CPU virtualization (for example, VT-x)
  - Extended/Nested Page Table support, also referred to as EPT, NPT, or Rapid Virtualization Indexing (RVI )
  - No-eXecute support, also referred to as NX, XD, No-Execute Memory Protection, No Execute Mode Mem Protection, Execute Disable, or Execute Bit Support

**SPARC and x86 based systems require the following:**

- A minimum of 8 GB of physical RAM
- The kernel zone brand software package `brand/brand-solaris-kz`.

  For information on obtaining and installing software packages, see Chapter 3, “Installing, Removing, and Updating Software Packages” in *Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.3*.
- To use Remote Administrative Daemon (RAD), the `rad-zonemgr` package must be installed on your system. For operations such as zone migration that occur between systems, the `rad-zonemgr` package must be installed on both the target and the source systems. Note that you must restart the RAD SMF services with the command `svcadm restart rad` after you install RAD modules.
- To prevent memory errors, you must adjust a parameter for the ZFS Adaptive Replacement Cache (ARC) on the kernel zone host. See “Tuning the ZFS ARC to Reserve Memory for Kernel Zones” on page 15.

Kernel zones can be installed using any of the following: the global zone's publishers and a default AI manifest, a custom AI manifest, an ISO image of Oracle Solaris installation media, or a Unified Archive.

The default AI manifest, `/usr/share/auto_install/manifest/default.xml`, and the global zone's `pkg` publishers are used to perform the installation unless the `-a`, `-b`, or `-m` options are specified. The supported installers are the text installer and the automated installer. This allows any supported Oracle Solaris version to be installed. Oracle Solaris 11.2 is the first version of Oracle Solaris supported in a kernel zone.

Oracle Solaris Zones can run in guests on Oracle VM Server for SPARC (previously called Sun Logical Domains). Each Oracle VM Server for SPARC domain has an independent limit for the number of kernel zones that you can run. The limit is 768 for SPARC T4 or SPARC T5 systems, 512 for SPARC M5 or SPARC M6 systems, and 256 for Fujitsu M10 systems.

Kernel zones cannot run in Oracle VM Server for x86 guests or on Oracle VM VirtualBox.

**Note -** On SPARC based systems, a running kernel zone within an Oracle VM Server for SPARC domain will block Oracle VM Server for SPARC live migration of the guest domain. Refer to the Oracle Solaris 11.3 release notes for further information.

Kernel zone live migration on SPARC based systems has additional software and firmware requirements. See "Live Migration Requirements" on page 54.

**Note -** Although you can run different zone brands on a system, when you run kernel zones you should reserve the kernel zone host for running zones and avoid running applications in the global zone.

## Verifying Hardware and Software Support on Kernel Zone Hosts

Before planning and deploying a kernel zone, you must verify that the kernel zone host has the hardware and software requirements as described in "Hardware and Software Requirements for Oracle Solaris Kernel Zones" on page 12. You can use the `virtinfo` command to verify the hardware requirements, firmware or BIOS requirements, and kernel zone brand package software requirements on the kernel zone host.

### ▼ How to Verify That a System Can Support Kernel Zones

1. **On the kernel zone host, become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Verify that the Oracle Solaris operating system version is at least 11.2.**

   ```
   # uname -a
   ```

   For example, on the system `global`:

   ```
   global# uname -a
   ```

```
SunOS global 5.11 11.2 sun4v sparc sun4v
```

3. **Verify the installation of the kernel zone brand package `brand/brand-solaris-kz`.**

   ```
   # pkg list brand/brand-solaris-kz
   ```

   The following example shows that the kernel zone brand package is installed on the system `global`.

   ```
   global#  pkg list brand/brand-solaris-kz
   NAME (PUBLISHER)                             VERSION                      IFO
   system/zones/brand/brand-solaris-kz          0.5.11-0.175.2.0.0.36.22321  i--
   ```

4. **Run the `virtinfo` command.**

   ```
   # virtinfo
   ```

   The following example output shows that kernel zones are supported on the system `global`, which is a logical domain.

   ```
   global# virtinfo
   NAME            CLASS
   logical-domain  current
   non-global-zone supported
   kernel-zone     supported
   ```

**See Also** For further information, see the `virtinfo`(1M) man page.

# Tuning the ZFS ARC to Reserve Memory for Kernel Zones

To ensure efficient performance of kernel zones, you must set the `user_reserve_hint_pct` tunable parameter on the system that is hosting kernel zones. The parameter provides a hint to the system about application memory usage and is used to limit growth of the ZFS Adaptive Replacement Cache (ARC) so that more memory remains available for applications. From the system perspective, a kernel zone itself is an application. Limiting the growth of the ARC ensures that more memory remains available for applications including the kernel zones and the applications running within them.

**Caution -** Failure to set this parameter to limit the host system's ZFS ARC can lead to low memory failures.

To limit the ZFS ARC on the system, as an administrator, set the `user_reserve_hint_pct` parameter in the global zone. The recommendation is to set the parameter value to `80` using a script called `set_user_reserve.sh` which adjusts the parameter dynamically on a running system.

You could set a value higher or lower than 80 depending on maximum memory requirements of all kernel zones and other processes that are anticipated to run on the host system.

To obtain the `set_user_reserve.sh` script and see more information about determining requirements and configuring the `user_reserve_hint_pct` tunable parameter, log in to the My Oracle Support website and access the document Memory Management Between ZFS and Applications in Oracle Solaris 11.x (Doc ID 1663862.1). The `set_user_reserve.sh` script is attached to that document.

Use the `set_user_reserve.sh` script to set the parameter. For example, in the global zone on the system named `global`:

```
global# ./set_user_reserve.sh -fp 80
Adjusting user_reserve_hint_pct from 0 to 80
Monday, March 30, 2015 04:59:47 PM PST :
waiting for current value : 60 to grow to target : 65
...
Adjustment of user_reserve_hint_pct to 80 successful.
Make the setting persistent across reboot by adding to /etc/system

#
# Tuning based on MOS note 1663861.1, script version 1.0
# added Monday, March 30, 2015 05:09:53 PM PST by system administrator : <me>
set user_reserve_hint_pct=80
```

Note that when you run the script, the `user_reserve_hint_pct` parameter is tuned on the running system, but you must set the parameter in `/etc/system` to make it persist across reboot.

## Configuring the Oracle Solaris Kernel Zone

This section describes how to configure an Oracle Solaris kernel zone.

### How to Configure a Kernel Zone

This procedure describes how to configure a kernel zone using the default kernel zone template, `SYSsolaris-kz`, which configures four virtual CPUs and four gigabytes of memory. An additional template `SYSsolaris-kz-minimal` is available to configure a minimal kernel zone with one virtual CPU and two gigabytes of memory.

For an overview of zone template properties, see "zonecfg template Property and Tokens" in *Oracle Solaris Zones Configuration Resources* . For general information regarding zone configuration, see Chapter 1, "How to Plan and Configure Non-Global Zones" in *Creating and Using Oracle Solaris Zones*.

**Before You Begin** You must confirm kernel zone hardware support, software support, and memory configuration on your host system. See "Verifying Hardware and Software Support on Kernel Zone Hosts" on page 14 and "Tuning the ZFS ARC to Reserve Memory for Kernel Zones" on page 15.

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Create a new kernel zone configuration.**

   The default `solaris-kz` branded zone template is `SYSsolaris-kz`. For example, on the system `global`, to create a new kernel zone configuration for the kernel zone, `kzone1`:

   ```
    global#  zonecfg -z kzone1
   Use 'create' to begin configuring a new zone.
   zonecfg:kzone1> create -t SYSsolaris-kz
   ```

   The remaining configuration steps in this procedure use the kernel zone `kzone1`.

3. **Add any additional kernel zone resources.**

   You can set some kernel zone resources now or after the zone is configured. For more information, see "Configuring and Customizing Kernel Zone Resources" on page 18.

4. **Commit the zone configuration.**

   ```
   zonecfg:kzone1> commit
   ```

5. **Exit `zonecfg`.**

   ```
   zonecfg:kzone1> exit
   ```

6. **(Optional) Verify the zone configuration.**

   You can verify a zone prior to installation. If you skip this step, the verification is performed automatically when you install the zone. See "(Optional) How to Verify a Configured Zone Before It Is Installed" in *Creating and Using Oracle Solaris Zones*.

   ```
   # zoneadm -z zonename verify
   ```

   For example, to verify the kernel zone `kzone1` on the system `global`:

   ```
   global# zoneadm -z kzone1 verify
   ```

   If you see an error message and the zone fails to verify, make the corrections specified in the message and try the command again. If no error messages are displayed, you can install the zone.

# Configuring and Customizing Kernel Zone Resources

Zones configuration resources enable you to manage the system resources for a zone. You specify resources when creating a zone configuration. Some resources are supported only for kernel zones or only for native zones.

This section describes how to configure resources to add additional support for the following components:

- Kernel zone CPUs. See "Managing Kernel Zone CPUs" on page 18 .
- Kernel zone memory. See "Managing Kernel Zone Memory" on page 20 .
- Kernel zone storage devices. See "Managing Kernel Zone Storage Devices and Boot Order" on page 23.
- Kernel zone network devices and network configuration. See "Managing Kernel Zone Network Devices and Configuration" on page 25.
- Kernel zone network virtualization. See "Managing Single-Root I/O NIC Virtualization on Kernel Zones" on page 27.
- Kernel zone verified boot. See "Using Verified Boot to Secure an Oracle Solaris Kernel Zone" on page 35.
- Kernel zone suspend resources. See "Configuring the `suspend` Resource" on page 34.

You use the `zonecfg` command on the global zone to set or to modify kernel zone resources.

**Note -** You must be the global administrator or a user with appropriate authorization in the global zone to use the `zonecfg` command.

See *Oracle Solaris Zones Configuration Resources* and the `solaris-kz`(5) man page for additional information about zone resources.

## Managing Kernel Zone CPUs

By default, a kernel zone is given four virtual CPUs upon creation. You can change the number of virtual CPUs using any of the following methods to configure the number of kernel zone CPUs:

- Adding and modifying the `dedicated-cpu` resource
- Adding and modifying the `virtual-cpu` resource
- Adding CPUs from an `anet` latency group

See Chapter 1, "How to Plan and Configure Non-Global Zones" in *Creating and Using Oracle Solaris Zones* for general information on how to set the `virtual-cpu` and `dedicated-cpu` zone resources.

## Adding the `dedicated-cpu` Resource

Configuring the `dedicated-cpu` resource property is recommended for best performance. Setting this value designates the kernel zone to only run on those selected CPUs. No other processes on the system can run on the CPUs that are dedicated to the kernel zone.

You can assign the CPU value in terms of available cores or processors. Use `psrinfo -vp` to obtain processor information on the system. For example, the following `psrinfo -vp` output shows that there are four available cores on the system `global`:

```
global# psrinfo -vp
The physical processor has 4 virtual processors (0-3)
  x86 (GenuineIntel 206D7 family 6 model 45 step 7 clock 2400 MHz)
        Intel(r) Xeon(r) CPU E5-2609 0 @ 2.40GHz
```

**Note -** By default, setting `dedicated-cpu:ncpus` does not provide any control over which of the system's CPUs are allocated. This can lead to inconsistent results if the system is rebooted. You should use `dedicated-cpu:cpus` to specify the exact CPU to use. For more information, see "dedicated-cpu Resource" in *Oracle Solaris Zones Configuration Resources*.

See Chapter 1, "How to Plan and Configure Non-Global Zones" in *Creating and Using Oracle Solaris Zones* for general information on the `dedicated-cpu` zone resource.

**EXAMPLE 1** Adding a Dedicated CPU to a Kernel Zone

This example shows how to add a dedicated CPU to the kernel zone `kzone1`.

```
global# zonecfg -z kzone1
zonecfg:kzone1> info dedicated-cpu
zonecfg:kzone1> add dedicated-cpu
zonecfg:kzone1:dedicated-cpu> set ncpus=8
zonecfg:kzone1:dedicated-cpu> end
zonecfg:kzone1> info dedicated-cpu
      ncpus: 8
zonecfg:kzone1> exit
```

## Adding the `virtual-cpu` Resource

The `virtual-cpu` resource specifies the number of virtualized CPUs visible to the kernel zone. On the host, virtualized CPUs share CPU time with other zones. Setting the `virtual-cpu` resource is beneficial for consolidation, but can affect system performance.

Note that if you have already defined the `dedicated-cpu` resource, the default number of virtual CPUs configured matches the lower value of the `ncpus` range inside the `dedicated-cpu` resource. If both resources exist, they are cross-checked for consistency. See the zonecfg(1M) man page for further information.

**EXAMPLE 2** Adding Additional Virtual CPUs to a Kernel Zone

This example shows how to add additional virtual CPUs to the kernel zone kzone1 using the virtual-cpu resource.

```
global# zonecfg -z kzone1
zonecfg:kzone1> info virtual-cpu
zonecfg:kzone1> add virtual-cpu
zonecfg:kzone1:virtual-cpu> set ncpus=8
zonecfg:kzone1:virtual-cpu> end
zonecfg:kzone1> info virtual-cpu
virtual-cpu:
        ncpus: 8
zonecfg:kzone1> exit
```

### Adding CPUs from a Latency Group

You can specify CPUs from a latency group. Specifying CPUs from a latency group can improve network performance if the latency group is the same as the underlying network device.

For more information on working with latency groups, see Chapter 2, “Creating and Managing Virtual Networks” in *Managing Network Virtualization and Network Resources in Oracle Solaris 11.3*.

## Managing Kernel Zone Memory

You must allocate a fixed amount of physical RAM to the kernel zone virtual platform. You can define this amount by setting the kernel zone capped-memory resource's physical property.

The physical memory assigned to a kernel zone is allocated in its entirety when the zone boots. The memory allocated is for the exclusive use of the kernel zone. Once a kernel zone is booted, all of the memory specified in the capped-memory resource appears to be in use to the host operating system.

The default kernel zone memory size (capped-memory:physical) is 4 gigabytes (GB). It is recommended that the memory size be increased to manage larger workloads.

The default CPU and memory configuration for kernel zones is 4 VCPUs and 4 GBs of memory, to facilitate running applications. An additional kernel zone template, SYSsolaris-kz-minimal, provides the minimal supported kernel zone configuration of 1 VCPU and 2 GBs of memory.

On an x86 based system, the capped-memory resource must be set in increments of 2 megabytes (MB).

On a SPARC based system, the capped-memory resource must be set in increments of 256 megabytes (MB).

The zone allocates the capped-memory resource when the zone boots. This amount remains fixed while the zone is running.

The capped-memory:pagesize-policy property specifies the policy for allocating page size for the kernel zone's physical memory. By default a kernel zone uses the largest page size available to enable best performance. See "About Memory Page Size Policy and Physical Memory" on page 22 for more information.

**Note -** The zone template SYSsolaris-kz-minimal provides the minimal supported kernel zone configuration of 1 VCPU and 2 GBs of memory. On the Fujitsu M10 or SPARC M10 servers, a kernel zone created with this template might not be bootable because of insufficient memory. If the kernel zone cannot be booted, increase the memory assigned to the kernel zone through the physical property of the capped-memory resource.

See Chapter 1, "How to Plan and Configure Non-Global Zones" in *Creating and Using Oracle Solaris Zones* for general information on how to set the capped-memory zone resource.

For detailed information on setting the capped-memory zone resource, see "solaris-kz Zones and the capped-memory Resource" in *Oracle Solaris Zones Configuration Resources*.

If you increase kernel zone memory size prior to installation, you must also increase the kernel zone root disk size to account for the larger swap and dump devices. If you do not explicitly add a disk to a kernel zone, a zvol is created and used as the root disk. By default, the zvol is 16 GB in size. If you require a different root disk size, use the zoneadm install -x install-size command to modify the disk size. For example, to specify a 32 GB root disk size on the kernel zone kzone1, you would use the following command when you install:

```
global# zoneadm -z kzone1 install -x install-size=32G
```

For information on modifying the disk size using the zoneadm command, see the zoneadm(1M) man page.

**EXAMPLE 3** Setting the capped-memory Resource on a SPARC Based System

This example shows how to specify 2048 MB of memory by setting the physical property of the capped-memory resource on a SPARC based system.

```
global# zonecfg -z kzone1
zonecfg:kzone1> select capped-memory
zonecfg:kzone1:capped-memory> set physical=2048m
zonecfg:kzone1:capped-memory> end
zonecfg:kzone1> exit
```

**EXAMPLE 4** Setting the capped-memory Resource on an x86 System

This example shows how to specify 16 GB of memory by setting the physical property of the capped-memory resource on an x86 system.

```
global# zonecfg -z kzone1
zonecfg:kzone1> select capped-memory
zonecfg:kzone1:capped-memory> set physical=16g
zonecfg:kzone1:capped-memory> end
zonecfg:kzone1> exit
```

## About Memory Page Size Policy and Physical Memory

The pagesize-policy property of the capped-memory resource controls how the system selects a page size for a kernel zone.

The default kernel zone template SYSsolaris-kz sets the pagesize-policy property to largest-available, which is the recommended value for best performance. This setting enables the system to select the appropriate page size to use with the kernel zone's amount of physical memory. The physical memory size must be a multiple of the page size, so the system selects the largest page size that aligns with the amount of physical memory specified for the kernel zone. Booting with pagesize-policy=largest-available always succeeds.

You can get best performance by setting an appropriate amount of physical memory to enable the largest page size to be selected when pagesize-policy=largest-available is set.

You must set the physical property to an amount that is a multiple of the largest page size supported. The amount of memory allocated must align perfectly with the page size being requested. See the following example.

**EXAMPLE 5** Setting Physical Memory to Use Largest Page Size

On a SPARC T5 system you can see in the output below that various page sizes are supported, The largest is 2147483648 bytes or 2GB.

To use the 2147483648 page size, the capped-memory:physical property is set to 8GB, a value that is a multiple of 2GB so the largest page size can be used when pagesize-policy=largest-available.

```
global# pagesize -a
8192
65536
4194304
268435456
2147483648
global# zonecfg -z kzone1
zonecfg:kzone1> select capped-memory
```

```
zonecfg:kzone1:capped-memory> set physical=8G
zonecfg:kzone1:capped-memory> info
capped-memory:
    physical: 8G
    pagesize-policy: largest-available
zonecfg:kzone1:capped-memory> end
zonecfg:kzone1> exit
```

If a kernel zone's `pagesize-policy` property is cleared or not set, the kernel zone uses the lowest allowable page size required to boot on the particular hardware platform on which it is running.

You must clear `pagesize-policy` if you want to migrate a kernel zone that was created in an update of Oracle Solaris 11.3 to an older Oracle Solaris release that does not support the `pagesize-policy` property, such as the initial release of Oracle Solaris 11.3. See the following example.

**EXAMPLE 6** Clearing the `pagesize-policy` Property Before Migration to Earlier Version of Oracle Solaris

This example shows how to clear the `pagesize-policy` property to prepare for migrating a kernel zone to an older release of Oracle Solaris that does not support the property.

```
global# zonecfg -z kzone1
zonecfg:kzone1> select capped-memory
zonecfg:kzone1:capped-memory> clear pagesize-policy
zonecfg:kzone1:capped-memory> end
zonecfg:kzone1> commit
```

You can do a trial run of the migration first to see if the `pagesize-policy` needs to be cleared using the `zoneadm migrate -n` option. See the `zoneadm`(8) man page or Chapter 3, "Migrating an Oracle Solaris Kernel Zone" for more information.

Rarely, you might want to set `pagesize-policy=largest-only` to prevent booting unless the largest page size is used. This might be useful for a kernel zone hosting a database or other application where performance depends on using the largest page size.

See "solaris-kz Zones and the capped-memory Resource" in *Oracle Solaris Zones Configuration Resources* for more information about setting `physical` and `pagesize-policy` properties.

## Managing Kernel Zone Storage Devices and Boot Order

A kernel zone root is always accessible. By default, a kernel zone installation uses a 16GB ZFS volume for the root disk. You can specify a different size at zone installation time by using the zoneadm -z install command with the -x install-size option. For example, to increase the ZFS size to 32 GB on the kernel zone kzone1:

```
global# zoneadm -z kzone1 install -x install-size=32g
```

You can add additional storage devices to a kernel zone by using the add device resource. Devices are portable across systems and provide increased performance over ZFS volumes. Additional kernel zone storage devices have the following requirements:

- The full storage device path (for example, /dev/rdsk/c9t0d0) must be specified.
- The storage device must be defined by only one of the following:
  - The add device match resource property. If you specify a storage device for the add device match resource property, you must specify a device that is present in /dev/rdsk, /dev/zvol/rdsk, or /dev/did/rdsk.
  - A valid storage URI.
- The storage device must be a whole disk or LUN.

Use the bootpri resource property to specify the boot order of each storage device. The bootpri resource property must be set to any positive integer value.

**Caution -** The bootpri resource property must be set only if the device is to be used as a boot device. If the bootpri resource property is set on devices other than boot devices, data corruption might result.

To unset the bootpri resource property, use the clear bootpri resource property.

If multiple bootable devices are present during installation, the devices will be used for a mirrored ZFS pool in the zone.

The default boot order of each device is determined by sorting devices first by bootpri, then by id if multiple devices have the same bootpri.

**EXAMPLE 7** Adding Additional Storage Devices to a Kernel Zone

This example shows how to add the additional storage device /dev/rdsk/c9t0d0 to the kernel zonekzone1.

```
global# zonecfg -z kzone1
zonecfg:kzone1> add device
zonecfg:kzone1:device> set match=/dev/rdsk/c9t0d0
zonecfg:kzone1:device> set bootpri=4
zonecfg:kzone1:device> end
```

**EXAMPLE 8** Changing the Kernel Zone Default Boot Device to Use a Storage URI:

This example shows how to change the default boot device on the kernel zone kzone1 to use a storage URI located at iscsi://zfssa/luname.naa.600144F0DBF8AF19000052E820D60003.

```
global# zonecfg -z kzone1
zonecfg:kzone1> select device id=0
zonecfg:kzone1:device> set storage=iscsi://zfssa/luname.naa.
600144F0DBF8AF19000053482CC00029
zonecfg:kzone1:device> end
zonecfg:kzone1> info device
device:
        match not specified
        storage: iscsi://zfssa/luname.naa.600144F0DBF8AF19000052E820D60003
        id: 0
        bootpri: 0
```

## Managing Kernel Zone Network Devices and Configuration

Kernel zones provide network access in kernel zones by adding net or anet resources. See "Configurable Resources and Properties" in *Oracle Solaris Zones Configuration ResourcesOracle Solaris Zones Configuration Resources* for more information about these two resource types.

**Note -** It is recommended to use an anet resource with kernel zones.

Exclusive-IP zones must be used for kernel zones. See "Exclusive-IP Zone Network Address" in *Creating and Using Oracle Solaris Zones* for more information about exclusive-IP zones.

You can supply additional MAC addresses to support nested zones, or zones where a kernel zone hosts non-global solaris and solaris10 branded zones. See "Managing Nested Zones" on page 68 for more information about nested zones.

You can optionally specify a network device ID to identify the VNIC address from inside the zone and determine the order in which the network interfaces are presented to the kernel zone. This process is similar to moving a NIC from one physical slot to another.

See Chapter 1, "How to Plan and Configure Non-Global Zones" in *Creating and Using Oracle Solaris Zones* for general information on how to set network zone resources.

**EXAMPLE 9** Adding Additional Network Resources to a Kernel Zone

This example shows how to add a network resource to the kernel zone kzone1. The ID is set to 3 to determine the order in which the new anet interface is presented to the kernel zone.

```
global# zonecfg -z kzone1
zonecfg:kzone1> add anet
zonecfg:kzone1:anet> set id=3
zonecfg:kzone1:anet> end
zonecfg:kzone1> exit
```

**EXAMPLE 10** Removing Network Devices From a Kernel Zone

This example shows how to remove a network device from the kernel zone `kzone1`. The information on the existing `anet` resources is listed and the `anet` device with the value of 1 is deleted.

```
global# zonecfg -z kzone1 info anet
anet:
        lower-link: auto
        allowed-address not specified
        allowed-dhcp-cids not specified
        link-protection: mac-nospoof
        mac-address: random
        mac-prefix not specified
        mac-slot not specified
        vlan-id not specified
        priority not specified
        rxrings not specified
        txrings not specified
        mtu not specified
        maxbw not specified
        rxfanout not specified
        vsi-typeid not specified
        vsi-vers not specified
        vsi-mgrid not specified
        etsbw-lcl not specified
        cos not specified
        id: 0
anet:
        lower-link: auto
        allowed-address not specified
        allowed-dhcp-cids not specified
        link-protection: mac-nospoof
        mac-address: default
        mac-prefix not specified
        mac-slot not specified
        vlan-id not specified
        priority not specified
        rxrings not specified
        txrings not specified
        mtu not specified
        maxbw not specified
        rxfanout not specified
        vsi-typeid not specified
        vsi-vers not specified
        vsi-mgrid not specified
        etsbw-lcl not specified
        cos not specified
        id: 1
global# zonecfg -z kzone1 remove anet id=1
```

## Managing Single-Root I/O NIC Virtualization on Kernel Zones

You can create and administer single root I/O (SR-IOV) NIC virtual functions (VF) on kernel zones by using the `zonecfg` `iov` `anet` property. SR-IOV enables the efficient sharing of Peripheral Component Interconnect Express (PCIe) devices among virtual machines and is implemented in the system hardware to achieve I/O performance that is comparable to native performance. For information on using SR-IOV in Oracle Solaris, see "Using Single Root I/O Virtualization With VNICs" in *Managing Network Virtualization and Network Resources in Oracle Solaris 11.3*.

The `zonecfg` `iov` property is only supported on kernel zones. No native `solaris` zone support is provided.

See "Resource Types and Properties" in *Oracle Solaris Zones Configuration Resources* for information on how to enable and configure the `zonecfg` `iov` `anet` property.

### How to Enable SR-IOV NIC Virtual Functions on a Kernel Zone With a Single `anet`

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Enable `iov` on an `anet`.**

   Using `zonecfg`, enable `iov` on a selected `anet`.

   ```
   # zonecfg -z kernel-zone
   zonecfg:kernel-zone> set lower-link=network-interface
   zonecfg:kernel-zone> select anet id=id-number
   zonecfg:kernel-zone:anet> set lower-link=network-interface
   zonecfg:kernel-zone:anet> set iov=iov-value
   zonecfg:kernel-zone:anet> end
   zonecfg:kernel-zone exit
   ```

The following example demonstrates enabling iov on an anet belonging to the kernel zone kzone1.

```
global# zonecfg -z kzone1
zonecfg:kzone1> select anet id=0
zonecfg:kzone1:anet> set lower-link=net1
zonecfg:kzone1:anet> set iov=auto
zonecfg:kzone1:anet> end
zonecfg:kzone1> exit
```

3. **(Optional) Confirm that the iov property is set for the anet in the kernel zone configuration.**

```
# zonecfg -z kernel-zone info anet id=id-number
```

For example, on the system global and the anet 0 of kernel zone kzone1:

```
# zonecfg -z kzone1 info anet id=0
anet:
        lower-link: net1
        allowed-address not specified
        configure-allowed-address: true
        ...
        iov: auto
        lro: auto
        id: 0
```

4. **Use dladm(1M) to ensure that SR-IOV is enabled on the chosen network interface.**

```
# dladm show-linkprop -p iov network-interface
```

For example, on the system global and the network interface net1:

```
global# dladm show-linkprop -p iov net1
LINK     PROPERTY        PERM VALUE        EFFECTIVE    DEFAULT   POSSIBLE
net1     iov             rw   on           on           auto      auto,on,off
```

5. **Boot the kernel zone.**

```
# zoneadm -z kernel-zone boot
```

For example, to boot the kernel zone kzone1 on the system global:

```
global# zoneadm -z kzone1 boot
```

6. **Verify that the VF was successfully added.**

```
# zlogin kernel-zone
kernel-zone# dladm show-phys
```

For example:

```
global# zlogin kzone1
kzone1# dladm show-phys
LINK    MEDIA       STATE    SPEED    DUPLEX    DEVICE
net0    Ethernet    down     0        unknown   ixgbevf0
```

**Example 11** Confirming the zonecfg iov Value on an anet

The following example shows the iov value on anet 0. The value is set to off, the default value.

```
global# zonecfg -z kzone1
zonecfg:kzone1> select anet id=0
zonecfg:kzone1:anet> info
anet:
        lower-link: net1
        allowed-address not specified
        configure-allowed-address: true
        ...
        iov: off
        lro: auto
        id: 0
zonecfg:kzone1:anet> end
zonecfg:kzone1> exit
```

**Example 12** Configuring iov and VLAN Tagging on an anet

The following example shows how to explicitly set a VLAN ID to enable VLAN tagging on an anet, which allows untagged and potentially malicious frames to be dropped.

```
global# zonecfg -z kzone1
zonecfg:kzone1> select anet id=0
zonecfg:kzone1:anet> set iov=auto
zonecfg:kzone1:anet> set vlan-id=11
zonecfg:kzone1:anet> end
zonecfg:kzone1> exit
```

## Using Virtual Functions and Shadow VNICs With Oracle Solaris Kernel Zones

A virtual function (VF) on a kernel zone is created when an anet belonging to a kernel zone is configured with the zonecfg iov property set to on or auto. The VF is assigned by the host system to the kernel zone.

Each VF assigned to a kernel zone has an associated shadow VNIC in the host. You can use shadow VNICs to show network statistics.

The following shows example output of the shadow VNIC `kzone1/net0` on the system `global`:

```
global# dladm show-link
LINK                CLASS     MTU    STATE    OVER
net1                phys      1500   unknown  --
net0                phys      1500   up       --
net2                phys      1500   up       --
kzone1/net0         vnic      1500   unknown  net1

global# dlstat show-link kzone1/net0
LINK               IPKTS   RBYTES    OPKTS   OBYTES
kzone1/net0        0        0        3      126
```

Because a shadow VNIC is unable to transfer data, you cannot use a shadow VNIC for DLMP or trunk aggregations. In addition, you cannot configure link properties on a shadow VNIC.

The `zonecfg` `anet` property `bwshare` enables a shadow VNIC to be set on a link only if the underlying physical link is supported. See the `dladm`(1M) and `zonecfg`(1M) man pages for additional information.

For additional information on VNICs and network configuration, consult *Managing Network Virtualization and Network Resources in Oracle Solaris 11.3*.

## Configuring Virtual LANs in Kernel Zones

Using Ethernet-based anets, you can create VNICs inside a kernel zone and configure them to be in their own virtual LAN (VLAN).

Use the `vlan` resource to add extra VLAN IDs (VIDs) to an existing `anet` resource to create new VLANs. See "Configurable Resources and Properties" in *Oracle Solaris Zones Configuration Resources* for more information about `anet` and `vlan` resources.

The `vlan` resource makes a kernel zone VLAN-aware. The host system forwards the packets meant for these VLANs without stripping the VLAN tag to the kernel zone. The kernel zone will then forward the packet to the right network client.

When transmitting data, packets from these VLANs are tagged by the kernel zone and passed on to the host. The host forwards the packets without stripping the tag, based on the destination MAC.

**Note -** It is not required to specify a `vlan-id` (known as the port VID or PVID) for an `anet` before you can add extra VLANs for an `anet`. If there is no PVID set, all the untagged packets that match the zone's MAC addresses are passed on to the zone from the host.

**EXAMPLE 13** Configuring a Kernel Zone with Additional VLANs

Configure a zone `kz0` with a mac-address of 0:1:2:3:4:5, PVID of 11, and two additional VIDs of 45 and 46.

```
global# zonecfg -z kz0
zonecfg:kz0> create -t SYSsolaris-kz
zonecfg:kz0> select anet id=0
zonecfg:kz0:anet> set mac-address=0:1:2:3:4:5
zonecfg:kz0:anet> set vlan-id=11
zonecfg:kz0:anet> add vlan
zonecfg:kz0:anet:vlan> set vlan-id=45
zonecfg:kz0:anet:vlan> end
zonecfg:kz0:anet> add vlan
zonecfg:kz0:anet:vlan> set vlan-id=46
zonecfg:kz0:anet:vlan> end
zonecfg:kz0:anet> info vlan
    vlan 0:
        vlan-id: 45
    vlan 1:
        vlan-id: 46
zonecfg:kz0:anet> end
zonecfg:kz0> commit
zonecfg:kz0> exit
```

After the zone is installed and booted, the `dladm show-vnic` command shows the following:

```
global# dladm show-vnic
LINK            OVER           SPEED  MACADDRESS    MACADDRTYPE IDS
kz0/net0        net4           10000  0:1:2:3:4:5   fixed       VID:11,45,46
```

The virtual-switch on the host system `global` is now configured to handle frames with following <mac-address, vlan-id> tuples:

- <0:1:2:3:4:5, 11>
- <0:1:2:3:4:5, 45>
- <0:1:2:3:4:5, 46>

Frames arriving with <0:1:2:3:4:5, 11> tuple have the VID stripped by the system `global` and passed on to the kernel zone `kz0`, so `kz0` never sees packets tagged with VID 11. Frames with <0:1:2:3:4:5, 45> and <0:1:2:3:4:5, 46> will be passed to `kz0` with their tags VID 45 and 46.

Inside `kz0`, if there is a VLAN datalink vlan45 with VID of 45, the virtual switch in `kz0` will strip VID 45 from the frame and pass the frame to vlan45. All the frames originating from vlan45 datalink inside `kz0` will be tagged by the virtual-switch in `kz0` and passed onto the anet in the host. The host anet will pass the frames directly to the NIC to be sent out.

**EXAMPLE 14** Display the List of VLAN IDs Supported in the Kernel Zone

Inside a kernel zone, use the `dladm show-phys -v` to determine the VLAN IDs that are supported on the physical datalinks.

```
global# zlogin kz0
kz0# dladm show-phys -v
LINK    VID   INUSE  CLIENT
net0    40    yes    vnic0,vnic1
        20    no     --
        15    yes    vnic2
net1    32    no     --
        11    no     --
        10    no     --
```

## Using Dynamic MAC Addresses and VLAN IDs in Kernel Zones

For most deployment cases, the MAC address and VLAN IDs used in a kernel zone can be statically configured before the zone is booted. However, in some cases you may not know ahead of time what values the kernel zone needs to use for MAC addresses and VLAN IDs of its VNICs. In this case you can specify prefixes of allowed MAC addresses and ranges of allowed VLAN IDs to enable the kernel zone to tell the host which MAC address and VLAN ID it needs to use when it boots. You can also enable the kernel zone to create a VNIC with any valid MAC address or VLAN ID.

**Note -** You should use the default static configuration when you know the number of MAC addresses and VLAN IDs and their values ahead of time. Static configuration is also required for SR-IOV VF based anets.

To enable dynamic configuration, set the `anet` properties `allowed-mac-address` and `allowed-vlan-ids` as shown in the following procedure.

For more information about these properties, see "Resource Type Properties" in *Oracle Solaris Zones Configuration Resources*.

### ▼ How to Use Dynamic MAC Addresses and VLAN IDs for Kernel Zone `anet` Configuration

1. **Become an administrator.**

For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Enable `allowed-mac-address` on an `anet`.**

   Using `zonecfg`, add an `anet` device and a `mac` resource and enable `allowed-mac-address` on it.

   ```
   # zonecfg -z kernel-zone
   zonecfg:kernel-zone> add anet
   zonecfg:kernel-zone:anet> add mac
   zonecfg:kernel-zone:anet:mac> add allowed-mac-address octet-prefix

   zonecfg:kernel-zone:anet:mac> end
   zonecfg:kernel-zone:anet>
   ```

3. **Enable `dynamic-vlan-id` on the `anet`.**

   Using `zonecfg`, add a `vlan` resource and enable `allowed-vlan-ids` on it.

   ```
   zonecfg:kernel-zone:anet> add vlan
   zonecfg:kernel-zone:anet:vlan> add allowed-vlan-ids id-range
   zonecfg:kernel-zone:anet:vlan> end
   zonecfg:kernel-zone:anet> end
   zonecfg:kernel-zone> exit
   ```

4. **Boot the kernel zone.**

   ```
   # zoneadm -z kernel-zone boot
   ```

5. **Login to the kernel zone.**

   ```
   # zlogin kernel-zone
   ```

6. **Verify in the kernel zone the dynamic addresses and IDs.**

   To determine which MAC prefixes and VLAN IDs are allowed, use the `dladm show-phys` command with the `-o` option:

   ```
   # dladm show-phys -o link,media,device,allowed-addresses,allowed-vids
   ```

   For example, to verify on a zone called `kzone1`:

   ```
   global# zlogin kzone1
   kzone1# dladm show-phys -o link,media,device,allowed-addresses,allowed-vids
   LINK   MEDIA       DEVICE   ALLOWED-ADDRESSES   ALLOWED-VIDS
   net0   Ethernet    zvnet0   fa:16:3f,           100-199,
                               fa:80:20:21:22      400-498,500
   ```

## Working with IPoIB and Kernel Zones

You can configure a kernel zone to support InfiniBand (IPoIB) devices by setting properties of the `anet` resource. Consult "Resource Types and Properties" in *Oracle Solaris Zones Configuration Resources* and "Creating and Viewing Paravirtualized IPoIB Datalinks in Kernel Zones" in *Managing Network Virtualization and Network Resources in Oracle Solaris 11.3*.

## Configuring the `suspend` Resource

Suspend and resume are supported for a kernel zone only if a kernel zone has a `suspend` resource property in its configuration. You must add the `suspend` resource and set its `path` or `storage` property before you can suspend the kernel zone.

Suspend and resume is necessary for warm migration. If you want to perform a warm migration, the `suspend` resource must use a shared storage location that is accessible to the source host and the target host.

Other uses for suspend and resume include enabling the ability to pause a zone instead of shutting it down when system maintenance is needed. Suspend and resume can enable the kernel zone and its running applications to be ready for use more quickly.

You can also set the `autoshutdown=suspend` property to enable a kernel zone to be suspended automatically instead of shut down when the global zone is shut down. See "Resource Types and Properties" in *Oracle Solaris Zones Configuration Resources* or `zonecfg`(1M) for more information about `autoshutdown`.

**EXAMPLE 15** Configure the suspend Resource to Enable Pausing a Kernel Zone

This example shows how to set the `suspend` resource to use a local path to enable suspend and resume so you can pause the kernel zone on the host.

```
global# zonecfg -z kz1
zonecfg:kz1> add suspend
zonecfg:kz1:suspend> set path=/system/zones/kz1/suspend
zonecfg:kz1:suspend> end
zonecfg:kz1> info suspend
suspend:
        path: /system/zones/kz1/suspend
zonecfg:kz1> exit
```

The zone can be suspended with the following command and later resumed with a `zoneadm boot` command.

```
global# zoneadm -z kz1 suspend
```

**EXAMPLE 16** Configure the suspend Resource to Enable Warm Migration

This example shows how to reset the suspend resource to use a storage URI for an iSCSI device.

```
global# zonecfg -z kz1
zonecfg:kz1> select suspend
zonecfg:kz1:suspend> clear path
zonecfg:kz1:suspend> set storage=iscsi://system/luname.naa.501337600144f0dbf8af1900
zonecfg:kz1:suspend> end
zonecfg:kz1> exit
```

See "Using Warm Migration to Migrate a Kernel Zone" on page 52 for more information.

See the solaris-kz(5) man page for more information on suspend resource property requirements.

## Using Verified Boot to Secure an Oracle Solaris Kernel Zone

You can use **verified boot** to secure a kernel zone's boot process. Verified boot protects a kernel zone from corrupted kernel zone modules, malicious programs, and installation of unauthorized third-party kernel modules by securely loading Oracle Solaris kernel modules before execution.

Verified boot enables you to perform the following actions:

- Automate the elfsign(1) verification of Oracle Solaris kernel modules. By default, you use only the Oracle Solaris system certificate for verification. With verified boot, you can specify additional certificates enabling you to load third-party kernel modules or modules signed for another version of Oracle Solaris.
- Create a verifiable chain of trust in the boot process beginning from kernel zone reboot up to the completion of the boot process.

Use the verified-boot zonecfg resource property to enable and to configure verified boot on a kernel zone.

Verified boot and the verified-boot resource property are supported only on solaris-kz brand zones.

For additional information about certificate verification and verified boot on Oracle Solaris 11.3, see the elfsign(1) man page and "Using Verified Boot" in *Securing Systems and Attached Devices in Oracle Solaris 11.3*.

## About the verified-boot Resource Property and elfsign Verification

The verified-boot resource property controls a kernel zone's boot policy and certificate settings. The properties of this resource are:

- policy

  The policy property regulates the verification of the unix, genunix and other kernel modules. The possible values for this property are as follows:

  warning — Prints a warning message if elfsign verification fails. This is the default value.

  none — No action occurs if elfsign signature verification fails.

  **Note -** The verified-boot resource property is not enabled if the policy value is set to none.

  enforce — Prints a warning message if elfsign signature verification fails. The kernel module does not load.

- cert

  The cert property specifies the location of the elfsign(1) X.509 public key certificate on the system. You specify the certificate location with a URI of the X.509 cert file. For a local file, the certificate must be located in the global zone's filesystem. For remote URIs, the URI must be accessible from the global zone.

  Use the add subcommand to add a certificate. You can add up to seven certificates on each kernel zone.

**EXAMPLE 17** Enabling Verified Boot in a Kernel Zone

This example creates the kernel zone kz1 on the system global. The verified-boot policy value is set to enforce. This directs the kernel to not boot if boot file signature verification fails and to print an error message on failure.

```
global# zonecfg -z kz1
kz1: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:kz1> create -t SYSsolaris-kz
zonecfg:kz1> set zonepath=/rpool/zones/kz1
zonecfg:kz1> set autoboot=true
zonecfg:kz1> add verified-boot
zonecfg:kz1:verified-boot> set policy=enforce
```

```
zonecfg:kz1:verified-boot> end
zonecfg:kz1> verify
zonecfg:kz1> commit
zonecfg:kz1> exit
```

**EXAMPLE 18** Configuring Kernel Zone Verified Boot With Multiple Certificates

This example demonstrates adding the `verified-boot` `zonecfg` resource to an already-configured kernel zone `kz2` on the system `global`. Two certificates are added to the configuration.

```
global# zonecfg -z kz2
zonecfg:kz2> add verified-boot
zonecfg:kz2:verified-boot> set policy=warning
zonecfg:kz2:verified-boot> add cert file:///etc/certs/SOLARIS-KZ
zonecfg:kz2:verified-boot> add cert http://example/keydist/cert.pem
zonecfg:kz2:verified-boot> info
verified-boot:
   policy: warning
    cert: file:///etc/certs/SOLARIS-KZ
    cert: http://example/keydist/cert.pem
zonecfg:kz2:verified-boot> end
zonecfg:kz2> verify
zonecfg:kz2> commit
```

# Enabling Silicon Secured Memory on a Kernel Zone

The Silicon Secured Memory (SSM) feature is available on SPARC based systems beginning with the SPARC T7 and SPARC M7 servers. SSM is sometimes called Application Data Integrity (ADI). To aid in migration of kernel zones to and from earlier systems, SSM is not enabled in a kernel zone by default even if SSM is available on the host system.

To enable SSM in a kernel zone, you can set the `host-compatible=adi` modifier. If you later migrate the kernel zone to an older SPARC based server or earlier version of Oracle Solaris where SSM is not available, you must set the property to a compatible value or clear it to enable the zone to work in the target system's environment.

The `host-compatible` modifier can be used to enable other release-specific features. See "solaris-kz SPARC Only: Kernel Zone Migration Class and Host Compatibility Level" in *Oracle Solaris Zones Configuration Resources* for more information.

**EXAMPLE 19** Enabling SSM in a Kernel Zone

This examples shows that `host-compatible` is not set, and then sets it to `adi` and boots the zone.

```
global# zonecfg -z kzone1
zonecfg:kzone1> info host-compatible
zonecfg:kzone1> set host-compatible=adi
zonecfg:kzone1> exit
global# zonecfg -z kzone1 boot
```

**EXAMPLE 20** Attempt to Enable SSM in a Kernel Zone on a System Without Silicon Secured Memory Feature

This example shows an attempt to enable SSM on a SPARC T5 system, which does not support SSM. The error is not detected until you boot the kernel zone.

```
global# zonecfg -z kzone1
zonecfg:kzone1> set host-compatible=adi
zonecfg:kzone1> exit
global# zonecfg -z kzone1 boot
zone 'kzone1': error: modifier adi not supported by migration class SPARC-T5
```

**EXAMPLE 21** Clearing `host-compatible` Property to Enable Migration to Earlier Systems

This example shows how to clear the `host-compatible` property so you can migrate a kernel zone to a target host that does not support features such as SSM. You must reboot after clearing.

```
global# zonecfg -z kzone1 clear host-compatible
zonecfg:kzone1> zoneadm -z kzone1 reboot
```

♦ ♦ ♦ CHAPTER 2

# Installing, Shutting Down, and Cloning Oracle Solaris Kernel Zones

This chapter describes how to install a kernel zone using several methods, how to uninstall a kernel zone, and how to halt, shut down, restart, and clone a kernel zone. This chapter includes the following topics:

- "Installing a Kernel Zone" on page 39
- "Uninstalling a Kernel Zone" on page 47
- "Shutting Down, Rebooting, and Halting a Kernel Zone" on page 48
- "Cloning a Kernel Zone" on page 48

For general information about zone installation and zone cloning concepts, see *Introduction to Oracle Solaris Zones*.

## Installing a Kernel Zone

Before you can install a kernel zone, you must configure it as described in Chapter 1, "Planning and Configuring Oracle Solaris Kernel Zones". After configuration, you install a kernel zone by using the `zoneadm install` command.

You can install a kernel zone through one of the following methods:

- A kernel zone direct installation. See "Installing a Kernel Zone by Using Direct Installation" on page 40.
- An Automated Installation (AI) manifest or an Oracle Solaris system configuration (`sysconfig`) profile. See "Using AI Manifests and `sysconfig` Profiles in Kernel Zone Installations" on page 42.
- Oracle Solaris installation image. See "Installing a Kernel Zone from an Installation Image" on page 46.

You can also clone a kernel zone that has already been installed. See "Cloning a Kernel Zone" on page 48.

## Installing a Kernel Zone by Using Direct Installation

Direct installation is the default kernel zone installation method. In a direct installation, the installer runs on the global zone. By default, the installer creates and formats the kernel zone boot disk and installs Oracle Solaris packages on that disk using the global zone's `pkg` publishers.

---

**Note -** In a kernel zone direct installation, the installer can recognize and install only the exact version of Oracle Solaris that is running on the global zone. To install a version of Oracle Solaris that is different from the version installed in the global zone, you must use an Automated Installation or interactive text installation. See "Installing a Kernel Zone from an Installation Image" on page 46.

---

A kernel zone direct installation occurs when you do not specify the `-b` option during a `zoneadm install` operation.

### ▼ How to Install a Kernel Zone Using Direct Installation

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Install the kernel zone.**

   ```
   # zoneadm -z zonename install
   ```

   For example, to install the kernel zone `kzone1` on the host system `global`:

   ```
   global# zoneadm -z kzone1 install
   ```

   ---

   **Note -** If a direct installation fails after zone verification, confirm that the publishers on the global zone have all of the required package components. See *Copying and Creating Package Repositories in Oracle Solaris 11.3* for more information.

   ---

3. **Boot the kernel zone.**

   ```
   # zoneadm -z zonename boot
   ```

   For example, to boot the kernel zone `kzone1` on the host system `global`:

   ```
   global# zoneadm -z kzone1 boot
   ```

4. **Log in to the kernel zone console to complete the zone configuration process.**

```
# zlogin -C zonename
```

For example, to log in to the console on the kernel zone `kzone1`:

```
global# zlogin -C kzone1
```

**Example 22** Installing a Kernel Zone Using Direct Installation

This example shows a successful direct installation of the kernel zone `kzone1`.

```
global# zoneadm -z kzone1 install
Progress being logged to /var/log/zones/zoneadm.20146T195713Z.kzone1.install
pkg cache: Using /var/pkg/publisher.
 Install Log: /system/volatile/install.778521/install_log
 AI Manifest: /tmp/zoneadm777933.spq5FV/devel-ai-manifest.xml
  SC Profile: /usr/share/auto_install/sc_profiles/enable_sci.xml
Installation: Starting ...

        Creating IPS image
         Startup: Retrieving catalog 'nightly' ... Done
         Startup: Caching catalogs ... Done
         Startup: Refreshing catalog 'nightly' ... Done
         Startup: Refreshing catalog 'solaris' ... Done
         Startup: Refreshing catalog 'extra' ... Done
         Startup: Caching catalogs ... Done
        Installing packages from:
            solaris
                origin:  http://ipkg.us.oracle.com/solaris11/dev/
         Startup: Linked image publisher check ... Startup: Refreshing catalog
 'nightly' ... Done
         Startup: Refreshing catalog 'solaris' ... Done
         Startup: Refreshing catalog 'extra' ... Done
        Planning: Solver setup ... Done
        Planning: Running solver ... Done
        Planning: Finding local manifests ... Done
        Planning: Fetching manifests:   0/477  0% complete
        Planning: Fetching manifests: 477/477  100% complete
        Planning: Package planning ... Done
        Planning: Merging actions ... Done
        Planning: Checking for conflicting actions ... Done
        Planning: Consolidating action changes ... Done
        Planning: Evaluating mediators ... Done
        Planning: Planning completed in 29.49 seconds
        The following licenses have been accepted and not displayed.
        Please review the licenses for the following packages post-install:
          consolidation/osnet/osnet-incorporation
        Package licenses may be viewed using the command:
          pkg info --license <pkg_fmri>

        Download:     0/52325 items    0.0/535.0MB  0% complete
        Download:  1024/52325 items   30.8/535.0MB  5% complete
        Download:  2233/52325 items   42.7/535.0MB  7% complete
        ...
        Download: 46744/52325 items  518.8/535.0MB  96% complete (6.4M/s)
        Download: Completed 534.98 MB in 79.80 seconds (5.0M/s)
         Actions:     1/74042 actions (Installing new actions)
         Actions: 17036/74042 actions (Installing new actions)
         ...
         Actions: 72796/74042 actions (Installing new actions)
         Actions: Completed 74042 actions in 97.96 seconds.
         Done
Installation: Succeeded
        Done: Installation completed in 359.901 seconds.
```

## Using AI Manifests and `sysconfig` Profiles in Kernel Zone Installations

You can use an Automated Installation (AI) manifest or `sysconfig` profile when you need to install multiple kernel zones with specific resource and package configurations different from that of the global zone.

Use the `zoneadm install` command with the `-m` option to specify an alternate AI manifest:

```
# zoneadm -z zonename install -m manifest
```

Use the `-c` option to specify a `sysconfig` profile:

```
# zoneadm -z zonename install -c sysconfig-profile
```

For example, to use the AI manifest `/data/archives/kzone-manifest.xml` to install the kernel zone `kzone1`:

```
global# zoneadm -z kzone1 install -m /data/archives/kzone-manifest.xml
```

Note the following guidelines when installing an alternate AI manifest or `sysconfig` profile to a kernel zone:

- For a successful installation, the AI manifest and `sysconfig` files must include the full path and `.xml` suffix.
- You cannot apply custom `disk` references in an AI manifest to a kernel zone installation. Because a kernel zone root disk is not available to the global zone, the kernel zone installation script automatically assigns a labeled loopback file, or `lofi`, device during configuration to allow for root disk creation. You can configure a removable loopback file `lofi` device, which works as a CD-ROM device, on the kernel zone. See "Managing Removable Devices on the Kernel Zone" on page 65.

  For additional information about developing and customizing an AI manifest, see Chapter 9, "Customizing Installations" in *Installing Oracle Solaris 11.3 Systems*. See "Zone

Installation and Administration Concepts” in *Creating and Using Oracle Solaris Zones* for further information about zone root disk creation.

- If you use an AI manifest to install a different version of Oracle Solaris than the one that is installed in the global zone, you must perform the installation from an image for the version of Oracle Solaris you are installing. See “Installing a Kernel Zone from an Installation Image” on page 46 for an example.

**EXAMPLE 23** Installing a Kernel Zone by Using a Separate Automated Installer (AI) Manifest

This example shows an installation of the kernel zone `kzone1` using the non-default Automated Install (AI) manifest `/var/tmp/kz_manifest.xml`.

```
global# zoneadm -z kzone1 install -m /var/tmp/kz_manifest.xml
Progress being logged to /var/log/zones/zoneadm.20146T195713Z.kzone1.install
pkg cache: Using /var/pkg/publisher.
 Install Log: /system/volatile/install.10708/install_log
 AI Manifest: /tmp/zoneadm10343.5la4Vu/devel-ai-manifest.xml
  SC Profile: /usr/share/auto_install/sc_profiles/enable_sci.xml
Installation: Starting ...

        Creating IPS image
         Startup: Retrieving catalog 'solaris' ... Done
         Startup: Caching catalogs ... Done
         Startup: Refreshing catalog 'solaris' ... Done
        Installing packages from:
            solaris
                origin:  http://pkg.oracle.com/solaris/release/
         Startup: Linked image publisher check ... Startup: Refreshing catalog
 'solaris' ... Done
        Planning: Solver setup ... Done
        Planning: Running solver ... Done
        Planning: Finding local manifests ... Done
        Planning: Fetching manifests:   0/501  0% complete
        Planning: Fetching manifests: 501/501  100% complete
        Planning: Package planning ... Done
        Planning: Merging actions ... Done
        Planning: Checking for conflicting actions ... Done
        Planning: Consolidating action changes ... Done
        Planning: Evaluating mediators ... Done
        Planning: Planning completed in 32.07 seconds
        The following licenses have been accepted and not displayed.
        Please review the licenses for the following packages post-install:
          consolidation/osnet/osnet-incorporation
        Package licenses may be viewed using the command:
          pkg info --license <pkg_fmri>

        Download:     0/64687 items    0.0/569.3MB  0% complete
        Download:   931/64687 items    5.8/569.3MB  1% complete (1.2M/s)
        ...
        Download: 64589/64687 items  569.2/569.3MB  99% complete (825k/s)
        Download: Completed 569.25 MB in 358.54 seconds (1.6M/s)
         Actions:     1/88614 actions (Installing new actions)
         Actions: 19471/88614 actions (Installing new actions)
         ...
         Actions: 86994/88614 actions (Installing new actions)
         Actions: 87128/88614 actions (Installing new actions)
         Actions: Completed 88614 actions in 73.71 seconds.
         Installation: Succeeded
         Done: Installation completed in 342.508 seconds.


Log saved in non-global zone as /zones/kzone1/root/var/log/zones/
zoneadm.20146T195713Z.kzone1.install
global#
```

**EXAMPLE 24** Installing a Kernel Zone Using Automated Installer (AI) Manifest for a Unified Archive (UAR) with Non-Root Pool

If a UAR contains datasets in a non-root pool and the AI manifest does not account for the non-root pool, you might see the following error:

```
ERROR: Archive contains non-root data, please use [-m manifest]
```

The following sample AI manifest is for installing from a Unified Archive (UAR) located at the path `/Extpool/Archive/Clone-T4.uar`. This archive was created on a system that has a non-root zpool named `tank`.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.1">
<auto_install>
  <ai_instance name="default">
    <target name="origin">
      <disk in_zpool="rpool" in_vdev="rpool-none" whole_disk="true">
        <disk_name name="c1d0" name_type="ctd"/>
      </disk>
      <disk in_zpool="tank" in_vdev="tank-none" whole_disk="true">
        <disk_name name="c1d1" name_type="ctd"/>
      </disk>
      <logical noswap="false" nodump="false">
        <zpool name="rpool" action="create" is_root="true"
            mountpoint="/rpool">
          <vdev name="rpool-none" redundancy="none"/>
        </zpool>
        <zpool name="tank" action="create" is_root="false"
            mountpoint="/tank">
          <vdev name="tank-none" redundancy="none"/>
        </zpool>
      </logical>
    </target>
    <software type="ARCHIVE">
```

```
      <source>
        <file uri="file:///Extpool/Archive/Clone-T4.uar"/>
      </source>
      <software_data action="install">
        <name>*</name>
      </software_data>
    </software>
  </ai_instance>
</auto_install>
```

If the manifest file is stored in `/tmp/ai.xml` and storage devices with id's 0 and 1 exist in the `kzone1` zone configuration, you could then use the following command to install in the kernel zone `kzone1`:

```
global# zoneadm -z kzone1 install -m /tmp/ai.xml
```

**EXAMPLE 25** Installing a Kernel Zone by Using an Alternate `sysconfig` Profile

This example installs the kernel zone `kzone1` using the `sysconfig` profile`/var/tmp/kzone1-sysconfig.xml`.

```
global# zoneadm -z kzone1 install -c /var/tmp/kzone1-sysconfig.xml
The following ZFS file system(s) have been created:
    rpool/zones/kzone1
Progress being logged to /var/log/zones/zoneadm.20146T195713Z.kzone1.install
pkg cache: Using /var/pkg/publisher.
AI Manifest: /tmp/zoneadm124827.zQWoOh/devel-ai-manifest.xml
SC Profile: /var/tmp/kzone1-sysconfig.xml
Installation: Starting ...

            Creating IPS image
             Startup: Retrieving catalog 'nightly' ... Done
             Startup: Caching catalogs ... Done
             Startup: Refreshing catalog 'nightly' ... Done
             Startup: Refreshing catalog 'solaris' ... Done
             Startup: Refreshing catalog 'extra' ... Done
             Startup: Caching catalogs ... Done
            Installing packages from:
                nightly
                    origin:  file:///server/nightly
                solaris
                    origin:  file:///server/solaris
                extra
                    origin:  http://server/extra
             Startup: Refreshing catalog 'nightly' ... Done
             Startup: Refreshing catalog 'solaris' ... Done
             Startup: Refreshing catalog 'extra' ... Done
            Planning: Solver setup ... Done
            Planning: Running solver ... Done
            Planning: Finding local manifests ... Done
            ...
            Planning: Fetching manifests: 552/552  100% complete
            Planning: Package planning ... Done
            Planning: Merging actions ... Done
            Planning: Checking for conflicting actions ... Done
            Planning: Consolidating action changes ... Done
            Planning: Evaluating mediators ... Done
            Planning: Planning completed in 56.62 seconds
            ...
            Download:  9746/65597 items  143.6/661.7MB  21% complete
            Download: 35018/65597 items  370.8/661.7MB  56% complete
            Download: 62181/65597 items  654.5/661.7MB  98% complete
            Download: Completed 661.67 MB in 40.57 seconds (0B/s)
             ...
             Actions: 87940/89672 actions (Installing new actions)
             Actions: 88107/89672 actions (Installing new actions)
             Actions: 88745/89672 actions (Installing new actions)
             Actions: Completed 89672 actions in 108.50 seconds.
             Done
Installation: Succeeded
        Done: Installation completed in 342.508 seconds.


Log saved in non-global zone as /zones/kzone1/root/var/log/zones/
zoneadm.20146T195713Z.kzone1.install
global#
```

## Installing a Kernel Zone from an Installation Image

You can install kernel zones from Oracle Solaris ISO installation images by using the `-b` option.

`# zoneadm -z` *zonename* `install -b` *path-to-iso-file*

Note the following considerations:

- Interactive text installations and automated installations from media are both supported. Live Media installation is not supported for kernel zones. See Part II, "Installing Using Installation Media," in *Installing Oracle Solaris 11.3 Systems* for additional information about these installation methods.
- The version of Oracle Solaris in the installation image must have support for kernel zones, so it must be at least Oracle Solaris 11.2. See "Verifying Hardware and Software Support on Kernel Zone Hosts" on page 14.
- To begin installation, you must include the complete path to the ISO image. Otherwise, the Oracle Solaris installation will fail.
- You can install from an installation image combined with an AI manifest with specific resource and package configurations.

During an Oracle Solaris installation from an ISO file, the kernel zone is booted and you are connected to the zone console. For information about how to use the zone console, see "Zone Console Login" in *Creating and Using Oracle Solaris Zones*.

**Caution -** If you exit or disconnect from the kernel zone console before the installation is complete,, the installation fails.

## ▼ How to Install a Kernel Zone From an Installation Image

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Install the kernel zone using the Oracle Solaris installation image.**

   ```
   global# zoneadm -z zonename install -b path-to-iso-file
   ```

   For example, to install the image located at `/var/tmp/solaris-media.iso` to the kernel zone, `kzone2`:

   ```
   global# zoneadm -z kzone2 install -b /var/tmp/solaris-media.iso
   ```

## ▼ How to Install a Kernel Zone From an Installation Image and Use an AI Manifest

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Install the kernel zone using the Oracle Solaris installation image and specify the manifest file with the `-m` option.**

   To install the image located at `/var/tmp/solaris-media.iso` to the kernel zone `kzone2` and also use an AI manifest `/var/tmp/kz_manifest.xml` with specific resource and package configurations:

   ```
   global# zoneadm -z kzone2 install -b /var/tmp/solaris-media.iso  -m /var/tmp/
   kz_manifest.xml
   ```

# Uninstalling a Kernel Zone

Use the `zoneadm uninstall` command to uninstall a kernel zone, for example, before you install a new or updated zone configuration. Note that the zone cannot be in the running state when you perform this operation. See "Shutting Down, Halting, Rebooting, and Uninstalling Zones" in *Creating and Using Oracle Solaris Zones* for information about zone uninstall procedures.

You must be the global administrator or a user with appropriate authorizations in the global zone to uninstall a zone.

## Shutting Down, Rebooting, and Halting a Kernel Zone

Use the `zoneadm shutdown`, `zoneadm reboot`, and `zoneadm halt` commands to shutdown, reboot, and halt a kernel zone. See "About Shutting Down, Halting, Rebooting, and Uninstalling Zones" in *Creating and Using Oracle Solaris Zones* for information about using these commands.

If you want a zone to boot automatically when the host system reboots, set the `autoboot` `zonecfg` resource. See Chapter 1, "How to Plan and Configure Non-Global Zones" in *Creating and Using Oracle Solaris Zones* for additional information about how to set this resource.

You must be the global administrator or a user with appropriate authorizations in the global zone to shut down, reboot, or halt a zone.

## Cloning a Kernel Zone

Cloning enables you to copy an existing configured and installed zone on your system to a new zone on the same system. The cloned zone includes any customizations of the existing zone. For example, added packages, modified zone resources, and file modifications on the source zone will appear in each cloned zone. Cloning a zone is an efficient way to add additional zones with a similar customized zone configuration.

You can clone a kernel zone in the following ways:

- Use the `zoneadm clone` command if you need to clone a small number of zones. See Example 26, "Cloning a Kernel Zone by Using the `zoneadm clone` Command.," on page 49.
- Use a Unified Archive file if you need to clone multiple zones for a large deployment, such as in a data center environment. See Example 27, "Cloning and Deploying a Kernel Zone by Using a Unified Archive," on page 49.

---

**Note -** A Unified Archive file can include only kernel zones that are in the running state. During Unified Archive creation, you can exclude any kernel zones that are not running. See Chapter 2, "Working With Unified Archives" in *Using Unified Archives for System Recovery and Cloning in Oracle Solaris 11.3* for more information.

---

After a kernel zone is cloned, you can boot and log in to the new zone.

**EXAMPLE 26** Cloning a Kernel Zone by Using the `zoneadm clone` Command.

This example demonstrates how to clone the kernel zone `kzone1` to the kernel zone `kzone2` on the host system `global`. For a step-by-step procedure, see "Cloning a Non-Global Zone on the Same System" in *Creating and Using Oracle Solaris Zones*.

```
global# zoneadm -z kzone1 halt
global# zonecfg -z kzone2 create -t kzone1
global# zoneadm -z kzone2 clone kzone1
Progress being logged to /var/log/zones/zoneadm.20140327T223951Z.kzone2.clone
Install Log: /system/volatile/install.100847/install_log
 AI Manifest: /system/shared/ai.xml
Installation: Starting ...

        Creating direct clone image...
        Registering dynamic archive transfer
        Pre-validating manifest targets before actual target selection
        Pre-validation of manifest targets completed
        Validating combined manifest and archive origin targets
        Commencing transfer of stream: ...
        Completed transfer of direct stream: ...
        Archive transfer completed
Installation: Succeeded
```

**EXAMPLE 27** Cloning and Deploying a Kernel Zone by Using a Unified Archive

The following example demonstrates cloning and deploying the kernel zone `kzone1` by using the `archiveadm` command. A Unified Archive is created for the kernel zone `kzone1`. The archive information is verified and the kernel zone `kzone2` is cloned with the modified zone configuration from `kzone1`. For a step-by-step procedure, please see *Using Unified Archives for System Recovery and Cloning in Oracle Solaris 11.3*.

```
global# archiveadm create -z kzone1 /var/tmp/kzone1.uar
Unified Archive initialized: /var/tmp/kzone1.uar.
                           \
Logging to: /system/volatile/archive_log.26248
Dataset discovery completed...
                            /
```

```
Media creation complete for zone(s)...
                             -
Archive stream creation completed...
                            -
Archive creation completed...
global# zoneadm list -cv
  ID NAME             STATUS      PATH                           BRAND      IP
   0 global           running     /                              solaris    shared
   2 kzone1           running     -                              solaris-kz excl
global# archiveadm info /var/tmp/kzone1.uar
Archive Information
          Creation Time:  2014-04-10T17:12:12Z
            Source Host:  global
           Architecture:  i386
       Operating System:  Oracle Solaris 11.2 X86
     Deployable Systems:  kzone1
global# zonecfg -z kzone2 create -a /var/tmp/kzone1.uar
global# zoneadm -z kzone2 install -a /var/tmp/kzone1.uar
global# zoneadm list -cv
  ID NAME             STATUS      PATH                           BRAND      IP
   0 global           running     /                              solaris    shared
   2 kzone1           running     -                              solaris-kz excl
   - kzone2           configured  -                              solaris-kz excl
```

♦ ♦ ♦ CHAPTER 3

# Migrating an Oracle Solaris Kernel Zone

A zone migration transfers an existing zone or system into a zone on another system. This chapter discusses kernel zone migration methods and administration.

## Kernel Zone Migration Requirements

Kernel zones share the same requirements for zone migration as other zone brands. These requirements are detailed in Chapter 7, "Migrating and Converting Oracle Solaris Zones" in *Creating and Using Oracle Solaris Zones*.

**Note -** Migrating a zone between SPARC and x86 architectures is not supported.

In addition, both the source and target hosts must have support for Oracle Solaris Kernel Zones. Refer to "Hardware and Software Requirements for Oracle Solaris Kernel Zones" on page 12.

**Note -** Kernel zone *live* migration has specific system firmware and storage requirements. See "Live Migration Requirements" on page 54 for details.

## Using Cold Migration to Migrate a Kernel Zone

You can use cold migration to migrate a kernel zone. In a cold migration, a zone is shut down, moved, and rebooted on another host. Cold migration is used for migrating applications that provide time-critical services or that have a large memory footprint.

Cold migration is supported by all zone brands. See Chapter 7, "Migrating and Converting Oracle Solaris Zones" in *Creating and Using Oracle Solaris Zones* for more information about cold migrations.

As part of a kernel zone cold migration, you must define the encryption key on the destination host. See "Encryption Keys And Kernel Zone Migration" on page 61.

# Using Warm Migration to Migrate a Kernel Zone

You can migrate a kernel zone to another host by suspending the zone on its current host using the `zoneadm suspend` command and resuming on a new host. This zone migration method is known as a *warm migration* or *migrating using suspend and resume*.

A warm migration does not require a full system reboot and restart of the application while the kernel zone is running.

Warm migrations require you to set up the zone configuration to be compatible on both the source and target hosts. For more information about zone configuration incompatibilities and warm migration, see the `solaris-kz`(5) man page.

Warm migrations require the zone to have a `suspend` resource configured for shared storage that is accessible by both the source and target hosts. See "Configuring the `suspend` Resource" on page 34 and Chapter 13, "Getting Started With Oracle Solaris Zones on Shared Storage" in *Creating and Using Oracle Solaris Zones*.

As part of a kernel zone warm migration, you must define the encryption key on the destination host. See "Encryption Keys And Kernel Zone Migration" on page 61.

## ▼ How to Migrate a Kernel Zone by Using Warm Migration

**Before You Begin** See "Configuring the `suspend` Resource" on page 34.

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Ensure that the kernel zone to be migrated has a `suspend` resource with shared storage configured.**

   For example:

   ```
   global# zonecfg -z kzone1 info suspend
   suspend:
           storage: iscsi://system/luname.naa.501337600144f0dbf8af1900
   ```

3. **On the global zone, suspend the file system on the kernel zone to be migrated.**

   ```
   # zoneadm -z zonename suspend
   ```

   For example, to suspend the kernel zone `kzone1` on the global zone `global`:

```
global# zoneadm -z zkone1 suspend
```

4. **Detach the kernel zone file system on the global zone.**

   ```
   # zoneadm -z zonename detach
   ```

   For example, to detach the kernel zone `kzone1` on the global zone `global`:

   ```
   global# zoneadm -z kzone1 detach
   ```

5. **Export the zone configuration and transfer the file to the target host.**

   This step will also configure a zone on the target host system using the same configuration as the source.

   ```
   # zonecfg -z zonename  export | ssh root@newhost zonecfg -z zonename  -f -
   ```

   For example:

   ```
   global# zonecfg -z kzone1 export | ssh root@global2 zonecfg -z kzone1 -f -
   ```

6. **Attach the zone on the new host.**

   ```
   # zoneadm -z zonename attach
   ```

   For example:

   ```
   global2# zoneadm -z kzone1 attach
   ```

7. **Boot the kernel zone on the new host to resume the migrated zone.**

   ```
   # zoneadm -z zonename boot
   ```

   For example:

   ```
   global2# zoneadm -z kone1 boot
   ```

## Using Live Migration to Migrate a Kernel Zone

Live migration enables you to migrate a kernel zone in the `running` state to a new kernel zone host. Because the memory state of a kernel zone is copied to the migrated guest, a live migration results in a brief outage time that is not noticeable to most applications or to most end users.

You can use live migration for any applications that require a minimum of downtime and where applications must remain in the `running` state.

## Live Migration Requirements

In addition to both the source and the target hosts meeting the minimum hardware and software requirements for kernel zones described in "Hardware and Software Requirements for Oracle Solaris Kernel Zones" on page 12, both source and target systems must meet the following requirements for live migration:

- The operating system on both hosts must be Oracle Solaris 11.3.
- If migrating between SPARC based systems, you must have the following firmware versions installed:
  - A SPARC T4 system with at least System Firmware 8.8
  - A SPARC T5, SPARC M5, or SPARC M6 system with at least System Firmware 9.5
  - A SPARC T7 or SPARC M7 series server. All firmware versions are supported.
  - A Fujitsu M10 or SPARC M10 server with at least XCP Firmware 2280.
- In addition to the firmware version requirements, if you are migrating between different SPARC architectures you must set the `cpu-arch` property as explained in "Specifying a CPU Migration Class for SPARC Kernel Zone Warm and Live Migration" on page 61.

Live migration source and target hosts must also have the following:

- All storage used by the zone must have shared storage accessible by both the source and target hosts. See Chapter 13, "Getting Started With Oracle Solaris Zones on Shared Storage" in *Creating and Using Oracle Solaris Zones*. Local-only disk paths are not supported on live migration.
- The zone configuration must be set up so that configuration is compatible and consistent on both the source and target hosts. See *Oracle Solaris Zones Configuration Resources*.
- Running instances of the following services:
  - The kernel zone live migration service, `svc:/network/kz-migr:stream`. Port 8102 on the target host must be open.
  - The Oracle Solaris Remote Administrative Daemon (RAD). TCP, TLS, and SSH transports are supported. Refer to the *Remote Administration Daemon Developer's Guide* for additional information on RAD.
  - The Network Time Protocol (NTP) server. See "Key Tasks for Managing Network Caching and Time-Related Services" in *Introduction to Oracle Solaris 11.3 Network Services* for additional information on NTP.

  For additional information on managing Oracle Solaris services, see Chapter 3, "Administering Services" in *Managing System Services in Oracle Solaris 11.3*.
- Configured SSH public key authentication between the source and target hosts. The public key authentication must be configured so that SSH does not require a prompt. See "How to Generate a Public/Private Key Pair for Use With Secure Shell" in *Managing Secure Shell Access in Oracle Solaris 11.3*.

It is recommended that a 10GB Ethernet link be used for kernel zone live migration.

## Authorizing Non-Root Users to Perform Kernel Zone Live Migration Operations

The `Zone Migration` rights profile enables a non-root user to use `zoneadm migrate` to migrate a kernel zone by using live migration.

The following output displays authorizations for a user `kz-user1` with the `Zone Migration` rights profile:

```
$ auths kz-user1
solaris.admin.wusb.read,solaris.mail.mailq,solaris.network.autoconf.read,solaris.zone.
migrate/vzl-112
$ profiles kz-user1
Zone Migration
Basic Solaris User
All
```

The user must also have `Zone Configuration` rights profile on the target system to create the configuration of the migrated zone.

For information about how to assign and manage rights profiles on Oracle Solaris, refer to *Securing Users and Processes in Oracle Solaris 11.3*.

## About the `zoneadm migrate` Command

Use the `zoneadm migrate` command to perform a live migration. `zoneadm migrate` command options include:

-n

Performs a dry run of the live migration. Confirms if a zone can be live migrated to the target host.

The zone is the `running` state.

-q

Quiet mode for live migrations. Specifies that the status is not reported during live migration operations.

-c *cipher*

Specifies a secure cipher option for live migrations. By default, migrations are secured using a cipher that is supported on both systems even if you do not specify a particular cipher. See "About Secure Live Migration" on page 59.

ssh|rads://*user*@*host*:*port*

Specifies a RAD URI including the scheme, user name, and host name to be used to migrate the zone to the target host. The `ssh` scheme uses SSH and the `rads` scheme uses TLS. If you only specify a host name, the scheme defaults to `rads`, *user* defaults to the current user, and *port* defaults to the standard RAD port 12302. See "Connecting to a RAD Instance by Using an URI" in *Remote Administration Daemon Developer's Guide* for more information.

Refer to the zoneadm(1M) man page for further information about the `migrate` command.

## ▼ How to Migrate a Kernel Zone By Using Live Migration

**Before You Begin** Ensure that both the kernel zone source and target hosts meet hardware, software, and storage requirements for live migration as detailed in "Live Migration Requirements" on page 54.

1. **Obtain administrative rights for live migration.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3* or "Authorizing Non-Root Users to Perform Kernel Zone Live Migration Operations" on page 55.

2. **Configure SSH authentication to not require an interactive prompt between the source and target hosts.**

   See "How to Generate a Public/Private Key Pair for Use With Secure Shell" in *Managing Secure Shell Access in Oracle Solaris 11.3*.

3. **Test SSH authentication by executing a command such as `date` on the target host.**

```
global1# ssh global2 date
Mon Mar  9 13:22:40 PDT 2015
```

   If you are prompted for a password, you have not configured your key pairs to enable login without interactive authentication.

   See "How to Generate a Public/Private Key Pair for Use With Secure Shell" in *Managing Secure Shell Access in Oracle Solaris 11.3*.

4. **Start the kernel zone migration service on the source and target hosts.**

```
global1# svcadm enable -rs svc:/network/kz-migr:stream
global1# ssh global2 svcadm enable -rs svc:/network/kz-migr:stream
```

5. **Check the status of RAD services on the source and target hosts.**

   The required services depend on which RAD URI you intend to use. If you will use `ssh` to migrate, the `svc:/system/rad:local` service must be running. If you will use `rads`, the `svc:/system/rad:remote` service must be running.

```
global1# svcs rad
```

```
STATE          STIME    FMRI
disabled       14:47:31 svc:/system/rad:remote
online         17:28:50 svc:/system/rad:local
online         17:28:56 svc:/system/rad:local-http

global1# ssh global2 svcs rad
STATE          STIME    FMRI
disabled       Jul_01   svc:/system/rad:remote
online         Jul_01   svc:/system/rad:local-http
online         Jul_01   svc:/system/rad:local
```

6. **(Optional) Start any disabled RAD services you require on the source and target hosts.**

   For example to enable the `svc:/system/rad:remote` service:

   ```
   global1# svcadm enable -rs svc:/system/rad:remote
   global1# ssh global2 svcadm enable -rs svc:/system/rad:remote
   ```

7. **Check the status of the NTP service on the source and target hosts.**

   ```
   global1# svcs ntp
   STATE          STIME    FMRI
   online         11:09:40 svc:/network/ntp:default

   global1# ssh global2 svcs ntp
   STATE          STIME    FMRI
   online         11:09:45 svc:/network/ntp:default
   ```

   If the NTP service is not online, see "Key Tasks for Managing Network Caching and Time-Related Services" in *Introduction to Oracle Solaris 11.3 Network Services* for information about how to set it up.

8. **On the source host, confirm that the zone to be migrated is in the `running` state.**

   ```
   global1# zoneadm list -cv
     ID NAME             STATUS      PATH      BRAND            IP
      0 global           running      /        solaris          shared
      1 kzone1           running      -        solaris-kz       excl
      - kzone2           installed    -        solaris-kz       excl
   ```

9. **Initiate a dry run or pre-flight check of the live migration.**

   On the source host, use `zoneadm migrate -n` to test the kernel zone configuration before a live migration.

   ```
   global1# zoneadm -z zonename migrate -n ssh://target-host
   ```

   For example, using the source host `global1` and the target host `global2`:

   ```
   root@global1:~# zoneadm -z kzone1 migrate -n ssh://global2
   zoneadm: zone 'kzone1': Importing zone configuration.
   ```

```
zoneadm: zone 'kzone1': Attaching zone.
zoneadm: zone 'kzone1': Booting zone in 'migrating-in' mode.
zoneadm: zone 'kzone1': Checking migration compatibility.
zoneadm: zone 'kzone1': Cleaning up.
zoneadm: zone 'kzone1': Dry-run migration successful.
```

10. **Migrate the kernel zone.**

    Migrate the kernel zone using `zoneadm migrate`.

    ```
    # zoneadm -z zonename migrate target
    ```

    For example, to migrate `kzone1` from the source host `global1` to the target host `global2`:

    ```
    root@global1:~# zoneadm -z kzone1 migrate ssh://global2
    zoneadm: zone 'kzone1': Importing zone configuration.
    zoneadm: zone 'kzone1': Attaching zone.
    zoneadm: zone 'kzone1': Booting zone in 'migrating-in' mode.
    zoneadm: zone 'kzone1': Checking migration compatibility.
    zoneadm: zone 'kzone1': Starting migration.
    zoneadm: zone 'kzone1': Suspending zone on source host.
    zoneadm: zone 'kzone1': Waiting for migration to complete.
    zoneadm: zone 'kzone1': Halting and detaching zone on source host.
    zoneadm: zone 'kzone1': Migration successful.
    ```

11. **Confirm that the zone has migrated and is now running on the target host.**

    ```
    root@global2:~# zoneadm list -cv
      ID NAME             STATUS      PATH        BRAND         IP
       0 global           running     /           solaris       shared
    1754 kzone1           running     -           solaris-kz    excl
       - kzone2           configured  -           solaris-kz    excl
       - kzone3           installed   -           solaris-kz    excl
       - kzone4           configured  -           solaris-kz    excl
    ```

12. **(Optional) On the source host, confirm that the zone that was migrated is in the `configured` state.**

    ```
    global1# zoneadm list -cv
      ID NAME             STATUS      PATH      BRAND            IP
       0 global           running      /        solaris          shared
       - kzone1           configured   -        solaris-kz       excl
       - kzone2           installed    -        solaris-kz       excl
    ```

    **Note -** If you later boot the source host into a boot environment (BE) that was created before the live migration, the state of the migrated kernel zone might show as `unavailable` instead of `configured` because the zone state is not shared across BEs. Issue the following command to detach the zone's storage and return the zone's state to `configured`.

    ```
    # zoneadm -z zonename detach -F
    ```

### **Example 28** Live Migration Pre-Flight Check Failure

This example demonstrates a failed pre-flight check between the source host `global1` and the target host `global2`. The `virtual-cpu` resource is inconsistent between both hosts. See *Oracle Solaris Zones Configuration Resources* for further information about zone configuration.

```
global1# zoneadm -z kzone1 migrate -n ssh://global2
zoneadm: zone 'kzone1': Using existing zone configuration on destination.
zoneadm: zone 'kzone1': Attaching zone.
zoneadm: zone 'kzone1': Booting zone in 'migrating-in' mode.
zoneadm: zone 'kzone1': boot failed:
zone 'kzone1': error: Suspended zone has 8 active VCPUs, more than the configured
zone 'kzone1': virtual-cpu maximum of 4.
zone 'kzone1': error: Correct errors, or delete the configuration, using zonecfg(1M) on
 the
zone 'kzone1': destination host.
zoneadm: zone kzone1: call to zoneadmd(1M) failed: zoneadmd(1M) returned an error 9
 (zone state change failed)
```

### **Example 29** Live Migration Between Hosts With Two Different `anet` Configurations

The following example demonstrates preparing for live migration between hosts with different `anet` configurations. The configuration for the zone `kzone1` on `global1` is not suitable on the target host `global2`. On `global2` the zone must use `net1` for its `anet` resource. The configuration is exported to the target host and modified, then the pre-flight check is performed.

See *Oracle Solaris Zones Configuration Resources* for additional information regarding `anet` resources.

```
global1# zonecfg -z kzone1 -r export | ssh root@global2 zonecfg -z kzone1 -f -
global1# ssh root@global2 zonecfg -z kzone1 'select anet 0; set lower-link=net1;end'
global1# zoneadm -z kzone1 migrate -n ssh://global2
```

## About Secure Live Migration

By default, live migration memory transfer data is encrypted when transferring between source and target hosts using an encryption cipher that is supported on both hosts. You can use `zoneadm migrate -c [`*cipher*`]` to specify a particular encryption cipher or disable encryption.

`zoneadm migrate -c` *cipher* has the following options:

none — Disables encryption

list — Lists supported ciphers on the source and target hosts.

*encryption-cipher* Specifies one of the ciphers that is supported on the source and target hosts. The `migrate -c list` command shows the possible values.

If you do not specify a cipher, one is automatically chosen based upon support of both the source and target hosts.

**EXAMPLE 30** Live Migration Between Two Trusted Hosts

The following example demonstrates a live migration of the kernel zone `kzone1` from the source host `global1` to the destination host `global2`. Encryption has been disabled.

```
global1# zoneadm -z kzone1 migrate -c none root@global2
    Password:
    zoneadm: zone 'kzone1': Using existing zone configuration on destination.
    zoneadm: zone 'kzone1': Attaching zone.
    zoneadm: zone 'kzone1': Booting zone in 'migrating-in' mode.
    zoneadm: zone 'kzone1': Checking migration compatibility.
    zoneadm: zone 'kzone1': Starting migration.
    zoneadm: zone 'kzone1': Suspending zone on source host.
    zoneadm: zone 'kzone1': Waiting for migration to complete.
    zoneadm: zone 'kzone1': Migration successful.
    zoneadm: zone 'kzone1': Halting and detaching zone on source host.
```

**EXAMPLE 31** Confirming Cipher Compatibility Between Live Migration Source and Destination Hosts

The following example demonstrates a live migration of the kernel zone `kzone1` from the source host `global1` to the destination host `global2`. The specified cipher `aes-128-cbc` is not supported on the destination host.

```
global1# zoneadm -z kzone1 migrate -c aes-128-cbc ssh://global2
zoneadm: zone 'kzone1': cipher aes-128-cbc not supported by destination
zoneadm: zone 'kzone1': destination supports: aes-128-ccm aes-128-gcm
```

**EXAMPLE 32** Listing Available Supported Ciphers on Live Migration Source and Destination Hosts

The following example lists the available supported ciphers during a live migration of the kernel zone `kzone1`. The zone is migrated from the source host `global1` to the destination host `global2`.

```
global1# zoneadm -z kzone1 migrate -c list root@global2
    Password:
    source ciphers: aes-128-ccm aes-128-gcm none
    destination ciphers: none
    # echo $?
    0
```

## Encryption Keys And Kernel Zone Migration

Kernel zones require encryption keys on the target host after a zone migration.

If you are migrating a zone by using live migration, the encryption key on the target host will be automatically defined as part of the live migration process.

If you are migrating a zone by means of a cold or a warm migration, use the `zonecfg export` command on the source system to generate a command file to be used on the target system. The command file will meet the requirements for a kernel zone encryption key. For example, to generate a command file for a zone migrated from `global1` to `global2`:

```
global1# zonecfg -z kzone1 export -f /net/.../kzone1.cfg
global2# zonecfg -z kzone1 -f /net/.../kzone1.cfg
```

## Specifying a CPU Migration Class for SPARC Kernel Zone Warm and Live Migration

For warm migration and live migration on SPARC based systems only, you can configure a kernel zone to have a CPU class that is different from the host system. You specify the kernel zone migration class for the CPU by using the `cpu-arch` `zonecfg` resource. See "solaris-kz SPARC Only: Kernel Zone Migration Class and Host Compatibility Level" in *Oracle Solaris Zones Configuration Resources* for further information about setting the `cpu-arch` `zonecfg` resource property for migration to a different CPU architecture.

If the kernel zone's CPU migration class is not set, the kernel zone's CPU migration class is the same as the host.

You must reboot the kernel zone to have the CPU migration class changes take effect.

**Note -** The kernel zone host will always refuse to resume a guest previously suspended on an incompatible platform. A kernel zone guest will not boot if the `cpu-arch` class is set to an incompatible value.

**EXAMPLE 33** Confirming and Setting the Kernel Zone Migration Class on a SPARC Based System

The below example demonstrates how to confirm and set the `cpu-arch` resource on the kernel zone `kzone1`.

```
global# zonecfg -z kzone1
zonecfg:kzone1> info cpu-arch
cpu-arch: generic
zonecfg:kzone1> set cpu-arch=migration-class1
```

```
zonecfg:kzone1> info cpu-arch
cpu-arch: migration-class1
zonecfg:kzone1> exit
```

**EXAMPLE 34** Live Migration Fails to Migrate Due to Incompatible CPU Architecture

This example demonstrates a live migration attempt between a SPARC T4 host `global1` and a SPARC T5 host, `global2`. The `cpu-arch` property is not consistent across the hosts and must be set as directed in "solaris-kz SPARC Only: Kernel Zone Migration Class and Host Compatibility Level" in *Oracle Solaris Zones Configuration Resources*.

```
global1# zoneadm -z kzone1 migrate -n ssh://global2
zoneadm: zone 'kzone1': Importing zone configuration.
zoneadm: zone 'kzone1': Attaching zone.
zoneadm: zone 'kzone1': Booting zone in 'migrating-in' mode.
zoneadm: zone 'kzone1': Checking migration compatibility.
zoneadm: zone 'kzone1': configuration check failed:
error: Suspended image cannot be resumed on current cpu migration class (SPARC-T4).
 Please check cpu-arch setting in zone config or in host LDom config.
2015-05-30 22:42:59 error: request failed: failed to create VM: Operation not supported
```

♦ ♦ ♦ CHAPTER 4

# Administering Oracle Solaris Kernel Zones

This chapter covers the following Oracle Solaris Kernel Zone administration topics:

For information about administrative topics for `solaris` and `solaris10` branded zones, see Chapter 9, "About Oracle Solaris Zones Administration" in *Creating and Using Oracle Solaris Zones*.

## Working in the Kernel Zone Environment

Working in a kernel zone environment is very similar to working in a global zone. This section describes the major differences between the kernel zone administrative environment and working with a global zone.

## Process ID Visibility in Zones

Kernel zone processes are not directly visible to the kernel zone host. You must use the `zlogin` command followed by a process management command to view any process information about a kernel zone. For example, to see process information about `syslogd` on the kernel zone `kzone1` from the kernel zone host `global`:

```
global# zlogin kzone1 ps -ef |grep syslogd
 root  1520     1   0 20:23:08 ?           0:00 /usr/sbin/syslogd
```

## Duplicate Process IDs in Kernel Zones

The global zone and each kernel zone manage their own process ID space. The same numeric process ID might identify different system processes in the global zone and in one or more kernel zones. For example, on the same system, you can have the numeric process 5678 running `syslogd` on the global zone and running `sendmail` on a kernel zone.

To kill process 5678 with the `ps` command in `kzone1`, use the `zlogin` command followed by the `kill` command.

```
global# zlogin kzone1 kill 5678
```

## Kernel Zone Zonepath

A kernel zone's zonepath, by design, cannot be set. It contains no persistent or otherwise serviceable data.

## Resource Management Functionality in Kernel Zones

Resource controls such as `max-processes` are not available when configuring a kernel zone. Because a kernel zone has an independent kernel from the global zone, a process running inside a kernel zone cannot take up a process table slot in the global zone.

# Working With Kernel Zones and Immutable Zones

Immutable Zones provide read-only, or immutable, file system profiles. Immutable zones are supported on both `solaris` branded zones (in non-global zones) and on kernel zones. For detailed information regarding immutable zones, see "Read-Only Zone Overview" in *Creating and Using Oracle Solaris Zones*.

# Managing Removable Devices on the Kernel Zone

You can configure a removable loopback file `lofi` device, which works as a virtual CD-ROM device, on the kernel zone.

## ▼ How to Add a Virtual CD-ROM Device to a Kernel Zone

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Create an empty removable read-only `lofi` device in the global zone.**

   ```
   # lofiadm -r
   ```

   The following example shows sample output.

   ```
   global# lofiadm -r
   /dev/lofi/1
   ```

3. **Add the `lofi` device to the kernel zone configuration.**

   ```
   # zonecfg -z zonename
   ```

   The following example demonstrates adding the `lofi` device located at `/dev/lofi/1` to the kernel zone `kzone1`:

   ```
   global# zonecfg -z kzone1
   zonecfg:kzone1> add device
   zonecfg:kzone1:device> set match=/dev/lofi/1
   zonecfg:kzone1:device> end
   zonecfg:kzone1> exit
   ```

4. **Reboot the kernel zone to have the configuration changes take effect.**

   ```
   # zoneadm -z zonename reboot
   ```

5. **Log in to the kernel zone.**

   ```
   # zlogin zonename
   ```

6. **On the kernel zone, update the device file system ( `devfs`) and restart the hardware abstraction layer (`hal`) so that the `hal` service will see the virtual `cdrom` device.**

```
zonename# devfsadm -i zvblk
zonename# svcadm restart hal
```

7. **List the removable devices on the kernel zone.**

```
zonename# rmformat -l
```

For example, the following example lists the removable devices on the kernel zone `kzone1`:

```
kzone1# rmformat -l
Looking for devices...
     1. Logical Node: /dev/rdsk/c1d0p0
        Physical Node: /zvnex/zvblk@0
        Connected Device: kz       vDisk            0
        Device Type: Removable
        Bus: <Unknown>
        Size: 16.4 GB
        Label: <Unknown>
        Access permissions: <Unknown>
     2. Logical Node: /dev/rdsk/c1d1p0
        Physical Node: /zvnex/zvblk@1
        Connected Device: kz       vCDROM           0
        Device Type: CD Reader
        Bus: <Unknown>
        Size: 0.0 MB
        Label: <Unknown>
        Access permissions: <Unknown>
```

8. **In the global zone, specify a path to an ISO image file to associate with the removable loopback device.**

```
global# lofiadm -r image-path device-path
```

The following example demonstrates associating the image path `/root/sol-11_3-repo.full.iso` with the `lofi` device `/dev/lofi/1`:

```
global# lofiadm -r /root/sol-11_3-repo-full.iso /dev/lofi/1
global# lofiadm
Block Device             File                           Options
/dev/lofi/1              /root/sol-11_3-repo-full.iso        Removable,Readonly
```

9. **Mount the CD-ROM device in the kernel zone.**

```
# mount -F hsfs device-location /mnt
```

The following example mounts the virtual CD-ROM device located at `/dev/dsk/c1d1p0`.

```
kzone1# mount -F hsfs /dev/dsk/c1d1p0 /mnt
```

10. **When you are finished using the virtual CD-ROM, unmount it from the mount point in the kernel zone.**

```
kzone1# umount /mnt
```

11. **Eject the CD-ROM virtual device in the kernel zone.**

```
kzone1# eject cdrom
```

12. **Verify that the ISO image is no longer associated with the lofi device in the global zone.**

```
# lofiadm
```

For example:

```
global# lofiadm
Block Device             File                           Options
/dev/lofi/1              -                              Removable,Readonly
```

## Working With Kernel Zone Auxiliary States

Kernel zones use *auxiliary states* to communicate supplementary state information to the global zone. A kernel zone does not have an auxiliary state set by default. Auxiliary states are set only when you initiate debugging and kernel maintenance operations.

To view the global zone current state and the kernel zone auxiliary states, use the `zoneadm list -s` command.

```
global# zoneadm list -s
NAME              STATUS           AUXILIARY STATE
global           running
kzone1           running
kzone2           running
kzone3           running            debugging
```

The kernel zone auxiliary states are as follows:

`suspended`

The zone has been suspended and will resume on the next boot. Note that the zone must be attached before this state is visible. A kernel zone appears in a `suspended` auxiliary state when undergoing a migration. See Chapter 3, "Migrating an Oracle Solaris Kernel Zone".

`debugging`

The kernel zone is in the kernel debugger, `kmdb`. Although the zone is in the `running` state, the zone cannot service any network requests. You must connect to the zone console to interact with `kmdb`. For information about how to connect to the zone console, see Chapter 4, "About Non-Global Zone Login" in *Creating and Using Oracle Solaris Zones*.

`panicked`
The zone is in the `running` state but has panicked. The host system is not affected. You must use zone console access to log in to a kernel zone in the panicked auxiliary state.

`migrating-out`
The zone is fully running, but is being migrated to another system.

`migrating-in`
The zone is booted on the system, and is receiving the migration image. It is not yet fully running until migration is complete.

For information about zone states, see Chapter 1, “Oracle Solaris Zones Introduction” in *Introduction to Oracle Solaris Zones*. For additional information about kernel zone auxiliary states, see the `solaris-kz`(5) man page. For information about the kernel debugger see the `kmdb`(1) man page.

## Managing Nested Zones

A *nested zone* is a non-global zone that is installed and booted from within a kernel zone. In nested zones, the kernel zone serves as the global zone. A nested zone can be a new `solaris` branded zone or a migrated `solaris` or `solaris10` branded zone. Kernel zones are not supported as a nested zone.

Nested zones have the following requirements:

**Operating System**
All nested zones must have support for Oracle Solaris 11.2 or higher.

- `solaris` branded zones running Oracle Solaris 11 or Oracle Solaris 11.1 must be updated to Oracle Solaris 11.2. See Chapter 3, “Installing and Updating Software Packages” in *Adding and Updating Software in Oracle Solaris 11.3* for information about updating system software packages.
- You can migrate a `solaris10` branded zone to a non-global zone running at least Oracle Solaris 11.2. See “Migrating a solaris10 Branded Zone to Another Host” in *Creating and Using Oracle Solaris 10 Zones* for procedures on how to migrate a `solaris10` zone to an Oracle Solaris 11.2 system.

**Network Configuration**
A `solaris` or `solaris10` branded zone that runs as a nested zone can use exclusive-IP or shared-IP. If you need an exclusive-IP configuration, you need to configure the kernel zone to allow for additional MAC addresses.

**System Resources**
Nested zones only can use system resources available to the kernel zone. These resources include virtual disks and iSCSI disks.

**Cloning**
If a kernel zone containing a nested configuration is cloned, only the outside kernel zone will be cloned. Any zones inside the kernel zone

are not cloned during the zone cloning process. See "Cloning a Kernel Zone" on page 48.

## How to Add Multiple MAC Addresses to a Kernel Zone

This procedure shows how to add two automatically generated MAC addresses to a kernel zone.

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Add the new MAC addresses.**

   ```
   global # zonecfg -z kz0
   zonecfg:kz0> add anet
   zonecfg:kz0:anet> add mac
   zonecfg:kz0:anet:mac> end
   zonecfg:kz0:anet> add mac
   zonecfg:kz0:anet:mac> end
   zonecfg:kz0:anet> end
   zonecfg:kz0> exit
   global #
   ```

3. **Boot the kernel zone, or apply the changes to the running kernel zone.**

   ```
   global # zoneadm -z kz0 apply
   zone 'kz0': Checking: Adding anet id=1
   zone 'kz0': Applying the changes
   ```

4. **(Optional) Log in to the kernel zone and display the new MAC addresses.**

   ```
   global # zlogin kz0
   kz0# dladm show-phys -m net1
   LINK                SLOT     ADDRESS            INUSE CLIENT
   net1                primary  2:8:20:42:cf:83    yes   net1
                       1        2:8:20:f4:e1:b1    no    --
                       2        2:8:20:38:67:f3    no    --
   ```

## Nested Zones and New Non-Global Zone Configuration

You can configure, install, and boot a new `solaris` branded zone from within a kernel zone using the `zonecfg` and `zoneadm` commands. For example:

```
kzone1# zonecfg -z zone1
Use 'create' to begin configuring a new zone.
zonecfg:zone1> create -t SYSsolaris
zonecfg:zone1> commit
zonecfg:zone1> exit
```

See *Creating and Using Oracle Solaris Zones* for additional information about planning, configuring, and installing non-global zones.

## Kernel Zone Host Data and Host ID

Each kernel zone bootable device contains state information known as *host data*. A kernel zone's host data monitors kernel zone state information including:

- Zone usage
- Zone suspends, as described in "Configuring the `suspend` Resource" on page 34
- Time of day offset between the kernel zone clock and the global zone clock
- OpenBoot variables (SPARC only)

Kernel zone host data is encrypted and authenticated with the advanced encryption standard AES-128-CCM, using the same encryption key used for the kernel zone `suspend` image.

When a kernel zone is configured or booted, the host data is read to determine whether the kernel zone's boot storage is in use on another system. If the boot storage is in use on another system, the kernel zone will enter the `unavailable` state and an error message will indicate which system is using the boot storage. For example:

```
global# zoneadm -z kzone1 attach
zone 'kzone1': error: ERROR: zone kzone1 is in use by host with  hostid 848611d4
zone 'kzone1': error:       last known state: installed
zone 'kzone1': error:               hostname: global2
zone 'kzone1': error:  boot environment name: solaris-1
zone 'kzone1': error:  boot environment uuid: 69ed2e6a-e25a-6d36-e022-ed7261ed8899
zone 'kzone1': error:       last update time: Sun Apr 13 20:08:13 2014
zone 'kzone1': error: To fix, detach the zone from the other host then attach it to this
 host
zone 'kzone1': error: If the zone is not active on another host, attach it with
zone 'kzone1': error:    zoneadm -z kzone1 attach -x force-takeover
```

If the boot storage is not in use by the other system, you can repair the kernel zone by using the `zoneadm attach -x force-takeover` command.

**Caution -** Forcing a takeover or reinitialization of the host data makes it impossible to detect if the zone is in use on any other system. Running multiple instances of a zone that reference the same storage leads to unrepairable corruption of the zone's file systems.

If a zone's encryption key is not accessible, the host data and any suspend image will not be readable. In such circumstances, any attempt to ready or boot the zone will cause the zone to enter the `unavailable` state. If recovery of the zone's encryption key is not possible, use the `zoneadm attach -x initialize-hostdata` command to generate a new encryption key and host data.

To prevent loss of the encryption key during a kernel zone migration, use the `zonecfg export` command on the source system to generate a command file to be used on the target system. For example:

```
global# zonecfg -z kzone1 export -f /net/.../kzone1.cfg
global# zonecfg -z kzone1 -f /net/.../kzone1.cfg
```

## Working With the Kernel Zone Boot Loader

The kernel zone boot loader manages booting operations on the kernel zone. To invoke the boot loader, the kernel zone must be in the ready or installed state. You can use the kernel zone boot loader to perform the following operations:

- List available boot environments
- Boot the zone to an alternate boot environment

Use the `zoneadm boot` command to invoke the kernel zone boot loader. You must also invoke the zone console when you invoke the kernel zone boot loader. The boot loader output will appear in the zone console.

**Note -** The command sequence to exit from the zone console is `~.`. See "How to Log In to the Zone Console" in *Creating and Using Oracle Solaris Zones* for additional information.

For information about creating and managing boot environments on the operating system level, see Chapter 1, "Introduction to Managing Boot Environments" in *Creating and Administering Oracle Solaris 11.3 Boot Environments*. Additional information for managing zones and boot environments is available in Chapter 2, "beadm Zones Support" in *Creating and Administering Oracle Solaris 11.3 Boot Environments*.

### ▼ How to Specify Alternate Boot Environments in a Kernel Zone

1. **Become an administrator.**

For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Log into the zone console.**

```
# zlogin -C zonename
```

For example, to log into the console on `kzone1`:

```
global# zlogin -C kzone1
```

3. **In a separate terminal window, list the available kernel zone boot environments.**

```
# zoneadm -z zonename boot -- -L
```

The following example shows sample output.

```
global# zoneadm -z kzone2 boot -- -L
[Connected to zone 'kzone2' console]
1 kz-130118 (rpool/ROOT/kz-130118)
2 kz-1 (rpool/ROOT/kz-1)
3 solaris-5 (rpool/ROOT/solaris-5)
4 solaris-7 (rpool/ROOT/solaris-7)
Select environment to boot: [ 1 - 4 ]:
```

4. **Boot to a selected boot environment.**

```
# zoneadm -z zonename boot -- -Z boot-environment
```

For example:

```
global# zoneadm -z kzone1 boot -- -Z rpool/ROOT/solaris-backup-1
```

**Example 35** Selecting and Booting Alternate Boot Environments on a SPARC Based System

The following example shows the zone console output for alternate boot environments for the kernel zone `kzone1`. The kernel zone host hardware is a SPARC based system.

```
[Connected to zone 'kzone1' console]
NOTICE: Entering OpenBoot.
NOTICE: Fetching Guest MD from HV.
NOTICE: Starting additional cpus.
NOTICE: Initializing LDC services.
NOTICE: Probing PCI devices.
NOTICE: Finished PCI probing.


SPARC T4-2, No Keyboard
Copyright (c) 1998, 2014, Oracle and/or its affiliates. All rights reserved.
```

```
OpenBoot 4.36.0.build_05, 2.0000 GB memory available, Serial #1845652596.
Ethernet address 0:0:0:0:0:0, Host ID: 6e026c74.



Boot device: disk0  File and args: -L
1 Oracle Solaris 11.2 SPARC
2 bootenv123
3 bootenv456
Select environment to boot: [ 1 - 3 ]: 2

To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/bootenv123

Program terminated
ok boot -Z rpool/ROOT/bootenv123

[NOTICE: Zone rebooting]
NOTICE: Entering OpenBoot.
NOTICE: Fetching Guest MD from HV.
NOTICE: Starting additional cpus.
NOTICE: Initializing LDC services.
NOTICE: Probing PCI devices.
NOTICE: Finished PCI probing.


SPARC T4-2, No Keyboard
Copyright (c) 1998, 2014, Oracle and/or its affiliates. All rights reserved.
OpenBoot 4.36.0.build_05, 2.0000 GB memory available, Serial #1845652596.
Ethernet address 0:0:0:0:0:0, Host ID: 6e026c74.

...
Hostname: kzone1
kzone1 console login:
```

### **Example 36** Selecting and Booting Alternate Boot Environments on an x86 Based System

The following example shows the zone console output for alternate boot environments for the kernel zone kzone1. The kernel zone host hardware is an x86 system.

```
[Connected to zone 'kzone1' console]
1 boot-2 (rpool/ROOT/boot-2)
2 Oracle Solaris 11.2 x86 (rpool/ROOT/solaris)
3 boot-1 (rpool/ROOT/boot-1)
Select environment to boot: [ 1 - 3 ]:2
Boot device: disk0  File and args:
reading module /platform/i86pc/amd64/boot_archive...done.
reading kernel file /platform/i86pc/kernel/amd64/unix...done.
SunOS global 5.11 11.2 i86pc i386 i86pc
Copyright (c) 1983, 2014, Oracle and/or its affiliates. All rights reserved.
Hostname: kzone1
```

```
...
kzone1 console login:
```

# Live Zone Reconfiguration in Kernel Zones

You can use live zone reconfiguration to reconfigure or to report on the live configuration of a `solaris-kz` zone while the zone is running. For more information on this feature, see Chapter 6, “Live Zone Reconfiguration” in *Creating and Using Oracle Solaris Zones*.

# NFS Storage URIs and Kernel Zones

You can configure an NFS Storage URI (Uniform Resource Identifier) for an Oracle Solaris kernel zone. Storage URIs are used to uniquely identify shared storage objects across different nodes. Shared storage enables you to transparently access and manage shared storage resources in zones.

NFS Storage URIs are only supported on kernel zones.

## NFS Storage URI Syntax and Usage

The NFS URI specifies an object-based on `lofi` device, created on the given NFS file. The NFS file is accessed with credentials derived from user and group. User and group can be given as user names or as user IDs. The host can be given as an IPv4 address, IPv6 address, or as a host name. IPv6 addresses must be enclosed in square brackets.

The *nfs-share-path* value must be an `nfs` export directory from the host server that contains a normal backing store file. NFS Storage URIs have the following syntax:

`nfs://`*user*`:` *group*@*host*`[:`*port*]/*nfs-share-path*/*file*

The following examples show how to use the URI syntax:

- `nfs://admin:staff@`*host*`/export/test/nfs_file`
- `nfs://admin:staff@`*host*`:1000/export/test/nfs_file`

NFS Storage URIs can be managed by the `suriadm` command. Use the `suriadm` property `mountpoint-prefix=/system/volatile/zones/`*zonename* for troubleshooting and recovery. See the `suriadm`(1M) man page or “Managing Storage URIs and Shared Storage Resources” in *Creating and Using Oracle Solaris Zones* for more information.

## Core Files in Kernel Zones

If a kernel zone process terminates abruptly, the resulting core file is saved on the kernel zone in a location defined by the `dumpadm` command.

A kernel zone may sometimes crash in conditions that prevent a core dump from generating within the kernel zone. To ensure that in such cases kernel zone core dumps are generated and accessible, use the `coreadm` command in the global zone to enable and to specify a location for these core dumps.

Refer to the `dumpadm` and `coreadm` man pages for additional information.

# Index