Creating and Using Oracle Solaris Kernel Zones

E61041-05
February 2024

Creating and Using Oracle Solaris Kernel Zones,

E61041-05

Primary Author: Cathleen Reiher, Sharon Veach, Raoul Carag

Contents

Index

Using This Documentation

- **Overview** – Describes how to plan, configure, install, and administer Oracle Solaris Kernel Zones
- **Audience** – Technicians, system administrators, and authorized service providers
- **Required knowledge** – Experience administering Oracle Solaris environments. Familiarity with Oracle Solaris Zones. 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=E37838_01.

Feedback

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

1

Planning Oracle Solaris Kernel Zones

This chapter discusses how to confirm the hardware, software, and security requirements for Oracle Solaris Kernel Zones, also known as `solaris-kz` branded zones. It provides procedures for verifying hardware and software support, and for configuring secure administration of kernel zones.

This chapter covers the following topics:

- About Oracle Solaris Kernel Zones
- Kernel Zones and General Zones Concepts
- Security Measures for Kernel Zones
- Software and Hardware Requirements for Oracle Solaris Kernel Zones
- SPARC: Software in Silicon Features on Kernel Zones

About Oracle Solaris Kernel Zones

An Oracle Solaris Kernel Zone, also called a `solaris-kz` zone, is a non-global zone that has its own kernel and operating system that are separate 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 of the global zone. For example, a kernel zone does not share software 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 a kernel zone. For more information, see Options That Operate on Non-Global Zones in *Updating Systems and Adding Software in Oracle Solaris 11.4*.

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 when the kernel zone is in the `configured` or `unavailable` state.

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

Kernel zones are part of the branded zones framework. For more information, see the `brands`(7) man page.

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

Caution:

On an Oracle Solaris x86 system, do not run Oracle VM VirtualBox and Oracle Solaris Kernel Zones at the same time.

Kernel Zones and General Zones Concepts

This guide 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.

Security Measures for Kernel Zones

Kernel zones, non-global zones, and global zones offer similar security features. IPsec and IKE protect the network, rights and auditing prevent unauthorized use of resources, and immutable zones add administrative security. For more information, see Security Measures for a System With Non-Global Zones in *Creating and Using Oracle Solaris Zones*.

Kernel zones has an additional security measure called **verified boot**. For more information, see Using Verified Boot to Secure an Oracle Solaris Kernel Zone and Using Verified Boot in *Securing Systems and Attached Devices in Oracle Solaris 11.4*.

The assignment of rights to administer zones to non-`root` users is a common security practice. By default, the global zone administrator (`root`) can administer all kernel zones, but `root` can distribute those rights.

For descriptions of zones rights profiles that apply to kernel zones and an overview of the `admin` resource in a zone, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

Note:

The examples and procedures in this guide assume that zones are administered by a non-`root` user. Typically, non-`root` users prefix zone administration commands with the `pfbash` or `pfexec` command to run the commands with rights. For more information, see the `pfexec`(1) man page.

Refer to the following for background and details about rights:

- Chapter 1, About Using Rights to Control Users and Processes in *Securing Users and Processes in Oracle Solaris 11.4*
- Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*
- admin Resource Type in *Oracle Solaris Zones Configuration Resources*

Software and Hardware Requirements for Oracle Solaris Kernel Zones

Oracle Solaris has Oracle Solaris release version requirements and hardware requirements for kernel zones. Additionally, to migrate zones requires hardware and software compatibility.

Software Requirements for Oracle Solaris Kernel Zones

To use an Oracle Solaris-supported feature within an Oracle Solaris kernel zone, both the OS of the host and the OS of its guest Kernel Zone must support the feature. Most Oracle Solaris features work out-of-box within a Kernel Zone.

Some features require additional support in the Kernel Zone hypervisor, in the Kernel Zone guest driver, or both. The Oracle Solaris system version of the host and the Kernel Zone guest determine whether a feature is supported from within a Kernel Zone. Some Oracle Solaris features are not yet supported from within a Kernel Zone.

Table 1-1 Kernel Zone Features and the Required Oracle Solaris Versions

Oracle Solaris kernel zone Feature	Kernel Zone Guest OS Versions	Host OS Versions
Software In Silicon	11.4, 11.3	11.4, 11.3
Live Zone Reconfiguration	11.4, 11.3	11.4, 11.3
Oracle Solaris Kernel Zone Migration	11.4, 11.3, 11.2	11.4, 11.3[1], 11.2[2]
Evacuation of Oracle Solaris Kernel Zone to Target Host	11.4, 11.3, 11.2	11.4

1 Adds Live Migration support.

2 Includes suspend and resume migration support.

Hardware Requirements for Oracle Solaris Kernel Zones

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 M8 series server. All firmware versions are supported.
- A SPARC T7 or SPARC M7 series server. All firmware versions are supported.
- A Fujitsu M10/SPARC M10 server. Follow the firmware requirements in *Fujitsu M10 Systems Product Notes* that are appropriate for your configuration.
- A Fujitsu SPARC M12 server with the Oracle Solaris 11.3 or Oracle Solaris 11.4 release. All firmware versions are supported.

For information about downloading the latest system firmware, see Firmware Downloads and Release History for Oracle Systems (https://www.oracle.com/servers/technologies/firmware/release-history-jsp.html).

x86 systems:

- Intel-based systems with Nehalem or later processors
- AMD-based systems with Barcelona or later processors
- Compatible microcode levels on the CPUs
- BIOS must enable the following:
  - 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

Both SPARC and x86 systems require the following:

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

  For information about obtaining and installing software packages, see Chapter 3, Installing and Updating Software Packages in *Updating Systems and Adding Software in Oracle Solaris 11.4*.
- To use the 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 the RAD SMF services must be manually restarted with the command `svcadm restart rad` after you install RAD modules.

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
- 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 text installer and the automated installer enable you to install any supported Oracle Solaris version. Oracle Solaris 11.2 is the first version of Oracle Solaris supported in a kernel zone.

Oracle Solaris Kernel 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 can run. The limit is 768 for SPARC T4 or SPARC T5 systems, 512 for SPARC M5, SPARC M6, SPARC M7, and SPARC M8 systems, and 256 for Fujitsu M10 and Fujitsu SPARC M12 systems.

Kernel zones cannot run in Oracle VM Server for x86 guests or on the 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. For details, see Kernel Zones Issues in *Oracle Solaris 11.4 Release Notes*.

Kernel zone live migration on SPARC based systems has additional software and firmware requirements. See Additional Requirements for Kernel Zone Warm Migration and Live Migration.

Tip:

Although you can run different zone brands on a system, when you run kernel zones, reserve the kernel zone host for running non-global 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 satisfies Software and Hardware Requirements for Oracle Solaris Kernel Zones.

How to Verify That a System Can Support Kernel Zones

1. 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 example-server 5.11 11.4.36.15.0 sun4v sparc sun4v non-virtualized
   ```

2. Verify that the kernel zone brand package brand/brand-solaris-kz is installed.

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

This 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         11.4-11.4.0.0.1.10.1       i--
```

3. Verify that the global zone supports kernel zones.

   ```
   $ virtinfo
   ```

   This sample output shows that kernel zones are supported on the system `global`.

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

For further information, see the `virtinfo`(8) man page.

SPARC: Software in Silicon Features on Kernel Zones

Oracle Software in Silicon features are available on servers based on Oracle's SPARC M7 processor and running Oracle Solaris 11.3 and Oracle Solaris 11.4. Software in Silicon technologies include Silicon Secured Memory (SSM) and data analytics accelerators (DAX). SSM enables Application Data Integrity (ADI).

For more information about Oracle Software in Silicon functionality, refer to Understanding the Security and Performance Advantages of a Complete Oracle Solution with Software in Silicon.

SSM and DAX are not enabled in a kernel zone by default, even if they are supported by the host system's CPU and the kernel zone's operating system. This default is chosen to aid warm and live migration of kernel zones to and from earlier systems which do not support these features.

To enable these features in a kernel zone running Oracle Solaris 11.3 or Oracle Solaris 11.4, set the `host-compatible` property:

- To enable only SSM, set `host-compatible=adi`.
- To enable DAX, virtual address masking (VA masking), and SSM, set `host-compatible=level1`.

Note:

Only features enabled by both migration class and host compatibility level are visible to the kernel zone. Do not set the `cpu-arch` property to a migration class if you want to use SSM or DAX.

To migrate a kernel zone to an earlier SPARC based system or earlier version of Oracle Solaris software where SSM or DAX is not available, *before* you begin the migration you must first make the following configuration changes:

1. Set the `host-compatible` property to a compatible value or clear the property to enable the zone to work in the target system's environment.
2. Set the `cpu-arch` property to migrate to an earlier SPARC based system. See Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions for more information.

See Kernel Zone Migration Class and Host Compatibility Level (solaris-kz Only) in *Oracle Solaris Zones Configuration Resources* for more information.

Example 1-1 Enabling SSM in a Kernel Zone

This example checks whether the `host-compatible` property is set, then sets the property to `adi` and boots the zone. Note that the `info` subcommand displays no information for a property that is not explicitly set, even when the property has a default value.

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

Example 1-2 Attempting to Enable SSM in a Kernel Zone on a System Without the Silicon Secured Memory Feature

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

```
global$ pfbash 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 1-3 Enabling DAX in a Kernel Zone

This example shows by the lack of output that the `host-compatible` property is not set on kernel zone `kzone1`, sets the property to `level1` to enable DAX, VA masking, and SSM, and boots the zone.

```
global$ pfbash zonecfg -z kzone1
zonecfg:kzone1> info host-compatible
zonecfg:kzone1> set host-compatible=level1
zonecfg:kzone1> exit
global$ zonecfg -z kzone1 boot
```

Example 1-4 Clearing the `host-compatible` Property to Enable Migration to Earlier Systems

This example clears the `host-compatible` property on kernel zone `kzone1` then reboots the zone. Note that you must also reset the `cpu-arch` property, as described in Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions, before you can migrate a kernel zone to a target host that does not support features such as SSM.

```
global$ pfbash zonecfg -z kzone1 clear host-compatible
global$ zoneadm -z kzone1 reboot
```

2

Configuring Oracle Solaris Kernel Zones

This chapter discusses how to configure Oracle Solaris Kernel Zones, also known as `solaris-kz` branded zones. The chapter covers the following topics:

- Configuring the Oracle Solaris Kernel Zone
- About Managing Kernel Zone Resources

Configuring the Oracle Solaris Kernel Zone

For an overview of zone template properties, see zonecfg Templates in *Oracle Solaris Zones Configuration Resources*. For general information regarding zone configuration, see Chapter 1, Before You Begin Working With Oracle Solaris Zones in *Creating and Using Oracle Solaris Zones*.

How to Configure a Kernel Zone

- Review requirements and guidelines for kernel zone configuration, including information about important kernel zone resources. See Planning Oracle Solaris Kernel Zones.
- Confirm kernel zone hardware support, software support, and memory configuration on your host system. See Verifying Hardware and Software Support on Kernel Zone Hosts.

Perform this procedure to configure a kernel zone that uses the default kernel zone template, `SYSsolaris-kz`.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Start the zonecfg utility and create a new kernel zone configuration.

   The default `solaris-kz` zone template is `SYSsolaris-kz`.

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

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 About Managing Kernel Zone Resources.

4. Commit the zone configuration.

   ```
   zonecfg:kzone> commit
   ```

5. Exit the zonecfg utility.

   ```
   zonecfg:kzone> exit
   ```

6. 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.

```
global$ zoneadm -z kzone 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.

After the kernel zone is verified, install and boot the kernel zone. Go to Installing a Kernel Zone.

About Managing Kernel Zone Resources

Zone configuration resources enable you to manage the system resources for a zone. You specify resources when creating a zone configuration. Some resources apply only to kernel zones.

This section describes how to manage resources for the following components:

- Managing Kernel Zone CPUs
- Managing Kernel Zone Memory
- Managing Kernel Zone Storage Devices and Boot Order
- Managing Kernel Zone Network Devices and Configuration
- Managing Single-Root I/O NIC Virtualization on Kernel Zones
- Configuring the suspend Resource Type
- Using Verified Boot to Secure an Oracle Solaris Kernel Zone
- Working with IPoIB and Kernel Zones

You use the `zonecfg` command in the global zone to manage kernel zone resources.

Note:

You must be an administrator assigned the appropriate zone administration rights in the global zone to use the `zonecfg` command. For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

For more information about zone resources, see *Oracle Solaris Zones Configuration Resources* and the `solaris-kz`(7) man page.

Managing Kernel Zone CPUs

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

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

Adding a `dedicated-cpu` Resource to a Kernel Zone

For best performance, configure a `dedicated-cpu` resource. By setting this value, you tell the kernel zone to run only 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 about the system. For example, the following `psrinfo -vp` output shows that there are six available cores on the system `global`:

```
global$ psrinfo -vp
The physical processor has 6 cores and 48 virtual processors (0-47)
The core has 8 virtual processors (0-7)
The core has 8 virtual processors (8-15)
The core has 8 virtual processors (16-23)
The core has 8 virtual processors (24-31)
The core has 8 virtual processors (32-39)
The core has 8 virtual processors (40-47)
SPARC-T4 (chipid 0, clock 2998 MHz)
```

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. For consistent results, see Allocating Dedicated CPUs to a Kernel Zone.

Example 2-1 Allocating Dedicated CPUs to a Kernel Zone

This example shows how to verify that no CPUs are allocated or dedicated to `kzone1`, and then allocates them. It assumes that `kzone1` is on the global system that is running a SPARC T4 with six cores.

```
global$ pfbash zonecfg -z kzone1
zonecfg:kzone1> info dedicated-cpu
zonecfg:kzone1> add dedicated-cpu
zonecfg:kzone1:dedicated-cpu> set cpus=8-15
zonecfg:kzone1:dedicated-cpu> end
zonecfg:kzone1> info dedicated-cpu
cpus:  8-15
zonecfg:kzone1:dedicated-cpu> end
zonecfg:kzone1> exit
```

For more information, see dedicated-cpu Resource Type in *Oracle Solaris Zones Configuration Resources*.

Adding a `virtual-cpu` Resource to a Kernel Zone

The `virtual-cpu` resource type specifies the number of virtualized CPUs visible to the kernel zone. On the host system, virtualized CPUs share CPU time with other zones. Setting a `virtual-cpu` resource is beneficial for consolidation, but can affect system performance. For more information and examples, see virtual-cpu Resource Type (solaris-kz Only) in *Oracle Solaris Zones Configuration Resources*.

Adding CPUs from a Locality Group to a Kernel Zone

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

For more information about working with locality groups, review the following:

- Chapter 2, Creating and Managing Virtual Networks in *Managing Network Virtualization and Network Resources in Oracle Solaris 11.4*
- *Best Practices for Oracle Solaris Kernel Zones* (https://www.oracle.com/technical-resources/articles/it-infrastructure/solaris-kernel-zones-best-practices.html)
- *Locality Group Observability on Oracle Solaris*

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 type'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. After a kernel zone is booted, all of the memory specified in the `capped-memory` resource is unavailable to the host operating system and other zones.

The default CPU and memory configuration for kernel zones is 4 VCPUs and 4 GB of memory, to enable applications to run in the kernel zone. Increase the kernel zone memory size (`capped-memory:physical`) to manage larger workloads:

- On a SPARC based system, set the resource in increments of 256 MBs.
- On an x86 based system, set the resource in increments of 2 MBs.

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 for more information.

An additional kernel zone template, `SYSsolaris-kz-minimal`, provides the minimal supported kernel zone configuration of 1 VCPU and 2 GB of memory.

Note:

On Fujitsu SPARC M12 servers, Fujitsu M10 servers, or SPARC M10 servers, a kernel zone created with the `SYSsolaris-kz-minimal` 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.

For detailed information about setting the `capped-memory` zone resource type, see Capped Memory Guidelines for a solaris-kz Zone 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. To specify a different root disk size, use the `zoneadm install -x install-size` command. For example, to specify a 32GB root disk size on the kernel zone `kzone1`, you would use the following command when you install:

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

For information about modifying the disk size by using the `zoneadm` command, see the `zoneadm`(8) man page.

Example 2-2 Setting the `capped-memory` Resource on a SPARC Based System

This example shows how to specify 2048MB of memory for a kernel zone on a SPARC based system.

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

Example 2-3 Setting the `capped-memory` Resource on an x86 Based System

This example shows how to specify 16GB of memory for a kernel zone on an x86 based system.

```
global$ pfbash zonecfg -z kzone1
zonecfg:kzone1> select capped-memory
zonecfg:kzone1:capped-memory> set physical=16g
zonecfg:kzone1:capped-memory> end ; 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.

For best performance, set an appropriate amount of physical memory that will enable the largest page size to be selected when `pagesize-policy=largest-available` is set.

If you have enabled memory live zone reconfiguration by setting the `host-compatible` property to the `native`, `level2`, or `memlzr` value, you can specify the `pagesize-policy` property to `fixed`. When `pagesize-policy=fixed`, you can use the `pagesize` property value to specify the exact page size that you want to allocate as kernel zone memory. The minimum page size is 2 Mbytes on x86 systems and 256 Mbytes on SPARC systems. Run the `pagesize -a` command to determine the available page size values on your platform.

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. This page size might not be appropriate. The `physical` property must be set to an amount that is a multiple of the largest page size supported, as shown in Setting Physical Memory to Use the Largest Page Size.

The amount of memory allocated must align perfectly with the page size being requested. Therefore, you must clear `pagesize-policy` if either of the following conditions apply:

- If the target system has a smaller page size than the source system.
- If the source kernel zone was created in an update of Oracle Solaris 11.4 and the target is an Oracle Solaris release that does not support the `pagesize-policy` property, such as the initial release of Oracle Solaris 11.3.

  See Clearing the pagesize-policy Property Before Migrating a Kernel Zone to an Earlier Oracle Solaris Release.

Use the `zonecfg` command to specify the page size policy of a kernel zone guest by setting the `capped-memory:pagesize-policy` property to one of the following valid values: `fixed`, `largest-available`, `largest-only`, and `smallest-only`. The default value is `largest-available`. See the `solaris-kz`(7) man page.

When `capped-memory:pagesize-policy=largest-available`, the operating system chooses the page size as follows:

- Selects the largest page size whose pages are available in a sufficient amount for the kernel zone to boot
- Ensures that the page size is aligned to the `capped-memory:physical` property value

Note that a kernel zone fails to boot if insufficient pages of the specified size are available or if the specified page size is invalid. Rather than falling back to the previous page size, the kernel zone fails to boot and issues an appropriate error message. See Kernel Zone Boot Failures.

If you specify `capped-memory:pagesize-policy=fixed`, you can specify any valid page size as the value of the `capped-memory:pagesize` property. Run the `pagesize -a` command to determine a valid page size value for your platform. The memory page size is the page size used by the address space of the kernel zone guest, not the page size of the guest system. The `capped-memory:pagesize` value is an integer with an optional scale unit (`K` for kilobytes, `M` for metabytes, `G` for gigabytes, and `T` for terabytes).

When you change the `capped-memory:pagesize-policy` property, the `capped-memory:pagesize` property, or both, reboot the kernel zone for the changes to take affect.

When you set the `capped-memory:pagesize-policy` and `capped-memory:pagesize` properties, you can update the kernel zone's memory (`capped-memory:physical`) in page-size chunks specified by `capped-memory:pagesize`. By specifying a smaller page size for your platform, you can increase or decrease the kernel zone's memory in a fine-grained way. The minimum memory page size for SPARC is 256 Mbytes and 2 Mbytes for x86. See Memory Live Zone Reconfiguration for Kernel Zones.

For example, you create a kernel zone with a memory size of 16 Gbytes that you might want to increase later to 20 Gbytes. If you specify `capped-memory:pagesize-policy=largest-only`, the page size is set to 16 Gbytes. This page size prevents you from increasing the memory size to 20 Gbytes because 20 Gbytes is not 16-Gbyte-aligned. Instead, specify `capped-memory:pagesize-policy=fixed` and specify a smaller valid page size, such as `2G` or `4G`, as the value of the `capped-memory:pagesize` property. Then, you can update the `capped-memory:physical` property value in page-size chunks specified by `capped-memory:pagesize`. So, setting `capped-memory:pagesize=2G` enables you to change the kernel zone's physical memory size in 2-Gbyte increments such as 12 Gbytes, 14 Gbytes, 18 Gbytes, 20 Gbytes, and so on.

Example 2-4 Setting Physical Memory to Use the Largest Page Size

The SPARC T5 supports various page sizes, as the following output shows. The largest is 2147483648 bytes or 2GB.

Using 2GB pagesize increments, the `capped-memory:physical` property is set to 8GB. This value enables the system to use the largest page size when `pagesize-policy` is set to `largest-available`.

```
global$ pfbash pagesize -a
8192
65536
4194304
268435456
2147483648
global$ pfbash 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 ; exit
global$ reboot
```

Example 2-5 Clearing the `pagesize-policy` Property Before Migrating a Kernel Zone to an Earlier Oracle Solaris Release

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

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

You can perform a trial run of the migration by using the `zoneadm migrate -n` command to see if the `pagesize-policy` needs to be cleared. For more information, see About the zoneadm migrate Command.

Tip:

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

For more information about setting the `physical` and `pagesize-policy` properties, see Capped Memory Guidelines for a solaris-kz Zone in *Oracle Solaris Zones Configuration Resources*.

Memory Reservation Pools for Kernel Zones

Oracle Solaris introduces memory resource pools (MRP) that can reserve system memory for use by kernel zones.

By default, Oracle Solaris global zones allocate memory for kernel zones from available system memory when a kernel zone boots. A kernel zone fails to boot if insufficient memory is available.

The recommended configuration uses an MRP to reserve memory for all kernel zones that are required to run in your production environment.

The MRP service management facility (SMF) service reserves memory for kernel zones. The reservation occurs early in the boot process of the global zone when system memory is readily available for reservation. To configure a kernel zone to allocate its memory from an MRP instead of from system memory. Use the `zonecfg` command to set the `capped-memory:memory-reserve` property. The default SMF service FMRI is `svc:/system/memory-reserve:zones`.

Memory alignment requirements apply to both the page size of an MRP and the memory size of a kernel zone that uses an MRP. If you want a kernel zone to use an existing MRP, the kernel zone's memory size must be aligned to the page size of the MRP. A kernel zone fails to boot if the memory is not aligned.

The page size that is used to populate an MRP is controlled by setting the `config/pagesize-policy` property in the MRP SMF service. Valid values are `smallest-only`, `largest-available`, `largest-only`, and `fixed`. If you choose the `fixed` value, you must set the `config/pagesize` property in the SMF service. See the `pagesize`(1) man page for the list of available page sizes on your platform.

For example, the system handles the `pagesize-policy=largest-available` setting for kernel zones and MRPs as follows:

- **Kernel Zone.** Based on a kernel zone's memory size, the system selects a suitable page size for the alignment. If a kernel zone's memory size and the memory page size that populate the MRP are aligned, a kernel zone that uses the MRP boots successfully. However, if these memory sizes are unaligned, the system adjusts the kernel zone's selected page size downward until it finds a valid alignment.
- **Memory Reservation Pool.** When you use `config/pagesize-policy=largest-available` to populate an MRP, only kernel zones that have memory sizes that align to the MRP's page size can boot successfully. So, a kernel zone boot might fail because of a memory alignment mismatch between the kernel zone and the MRP.

For example, you have an MRP that is populated with 2-Gbyte pages. If you configure a kernel zone with 5 Gbytes of memory to use that MRP, the memory-alignment mismatch causes the kernel zone boot to fail.

Enabling, disabling, and refreshing the `svc:/system/memory-reserve:zones` MRP service result in the following behavior:

- Enabling the service creates a memory reservation in an MRP. If an error occurs or the reservation fails, the service goes into maintenance modes and it logs a message that describes the reason for the failure.

- Disabling the service destroys the reservation in the MRP unless a kernel zone is using memory from the pool. If you attempt to disable the service while reserved memory is in use, the service instance drops into maintenance mode and it logs a message that describes the reason for the failure.
- Refreshing the service modifies the reservation of the MRP. After you configure an MRP, you can adjust the reservation size and refresh the service to increase or decrease the amount of memory in the pool. If the service fails to adjust the reservation size, an error message is logged and the MRP service is placed in the degraded state.

Note that using memory from an MRP for a kernel zone might cause the live migration of such a kernel zone to fail in the following circumstances:

- The MRP service instance specified by the `capped-memory:memory-reserve` property is disabled on the destination host.
- The page size of the MRP service on the destination host cannot be smaller than the page size on the source host.

Be sure to perform a live migration dry-run to detect potential migration failures.

Configuring a Memory Reservation Pool for Kernel Zones

An MRP is managed by the `svc:/system/memory-reserve:zones` service on both x86 and SPARC systems.

Use the `svccfg` SMF command to configure an MRP. Specify values for each of the following `svc:/system/memory-reserve:zones` service instance properties:

- `config/size` specifies the amount of memory to reserve for the MRP.

  For example, specify a value of `64G` to reserve (allocate) 64 Gbytes to the MRP.
- `config/pagesize-policy` mimics the `zonecfg capped-memory:pagesize-policy` property where valid values are `fixed`, `largest-available`, `largest-only`, and `smallest-only`.

  The default value is `largest-available`. This property value interacts with any kernel zone that uses the pool.
- `config/type` specifies the MRP type. The valid value for an MRP is `solaris-kz`, which is the default value.
- `config/lgrps` specifies the locality group (LGRP) ID or IDs to use. By default, this property specifies the root LGRP (`lgrp 0`), which spreads the reservation across all available LGRPs. The default property value is `""`.
- **SPARC only.** `config/npt-reservation` specifies the number of kernel zones that use this MRP instance simultaneously. Based on this property value, the MRP reserves *npt-reservation-value* x 16 Mbytes of memory.

By default, you configure a host to have a single MRP from which all kernel zones allocate memory.

On SPARC systems, configuring a kernel zone to use an MRP might require that you allocate NPT memory on a per-kernel-zone basis. This NPT memory is shared between the kernel zone virtual machine manager (ZVMM) and the SPARC hypervisor. To ensure that sufficient memory is available to make the NPT allocation for the kernel zone on a SPARC system, set the `config/npt-reservation` property to the number of kernel zones that use the particular MRP instance.

If an NPT memory allocation from the MRP fails, the ZVMM allocates NPT from system memory. If that allocation fails, an error message instructs you to either specify the number of kernel zones as the value to the `config/npt-reservation` property or to increase the value of the `config/npt-reservation` property for the MRP service. If a requested adjustment to the `config/npt-reservation` property is not possible, the kernel zone MRP service transitions to the degraded state and logs a message to the MRP service log.

When you view memory information by running the `mrpstat` command, the values in the `TOTAL_MEM`, `USED_MEM`, and `FREE_MEM` columns include the amount of NPT memory. See the `mrpstat`(8) man page.

The following example shows how to configure an MRP and reserve 64 Gbytes of `largest-available` page size of memory for use by kernel zones:

```
# svccfg -s svc:/system/memory-reserve:zones setprop config/size=64g
# svccfg -s svc:/system/memory-reserve:zones setprop config/pagesize-
policy=largest-available
# svccfg -s svc:/system/memory-reserve:zones setprop config/type=solaris-kz
# svccfg -s svc:/system/memory-reserve:zones setprop config/lgrps=""
# svccfg -s svc:/system/memory-reserve:zones refresh
# svcadm enable svc:/system/memory-reserve:zones
```

The following example helps to guarantee that four SPARC kernel zones can boot and reboot during the life cycle of a kernel zone host. Set the MRP `config/npt-reservation` service instance property value to `4`. Setting this property is in addition to setting the `config/size` MRP property to the cumulative memory size of the four kernel zones. Each kernel zone requires 16 Mbytes of capped memory. The MRP contains 64 Gbytes (4x16G) of memory plus the 64 Mbytes (4x16M) of NPT memory for the kernel zones.

```
# svccfg -s svc:/system/memory-reserve:zones setprop config/size=64G
# svccfg -s svc:/system/memory-reserve:zones setprop config/pagesize-
policy=largest-available
# svccfg -s svc:/system/memory-reserve:zones setprop config/npt-reservation=4
# svccfg -s svc:/system/memory-reserve:zones refresh
# svcadm enable svc:/system/memory-reserve:zones
```

Configuring a Kernel Zone to Use a Memory Reservation Pool

Configure a kernel zone to use an MRP by setting the `capped-memory:memory-reserve` property. A valid property value is the instance name of a specific MRP service. Only the `svc:/system/memory-reserve:zones` service instance of the `svc:/system/memory-reserve` SMF service is defined by default. This service instance is disabled by default. When you set the `capped-memory:memory-reserve` property, an SMF dependency is set on the corresponding `svc:/system/zones/zone` restarter service instance for the MRP service instance.

Note that the `memory-reserve` property is mutually exclusive with the `pagesize-policy` property. So, if you configure an MRP service instance, the kernel zone uses the page size of the MRP as specified by the MRP service's `pagesize-policy` property. If you clear the `memory-reserve` property, the kernel zone allocates from general system memory.

You cannot reconfigure the `capped-memory:memory-reserve` property dynamically. If you change this property value, you must reboot the kernel zone to make the changes take affect. If you change the page-size policy, you must reboot the kernel zone.

The following example commands configure the kz1 kernel zone to use 16 Gbytes of memory from the zones MRP:

```
# zonecfg -z kz1
zonecfg:kz1> select capped-memory
zonecfg:kz1:capped-memory> clear pagesize-policy
zonecfg:kz1:capped-memory> set memory-reserve=zones
zonecfg:kz1:capped-memory> info
capped-memory:
physical: 16G
memory-reserve: zones
zonecfg:kz1:capped-memory> end
zonecfg:kz1> commit
zonecfg:kz1> exit
```

Managing Kernel Zone Storage Devices and Boot Order

The root of a kernel zone is always accessible and by default is a 16GB ZFS volume. To increase the disk space, you can enlarge the root disk as described in Managing Kernel Zone Memory. Or, you can add storage devices. 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/dsk/c9t0d0) must be specified.
- The storage device must be defined by a valid storage URI.
- The storage device must be a whole disk or LUN.

Use the bootpri property of the device resource type to specify the boot order of each storage device. Set the bootpri property to any positive integer value.

Caution:

The bootpri property must be set only on a boot device. If the bootpri property is set on devices other than boot devices, data corruption might result.

To unset the bootpri property, use the zonecfg clear bootpri command.

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

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

Example 2-6 Adding Storage Devices to a Kernel Zone

This example shows how to add the storage device /dev/dsk/c9t0d0 to the kernel zone kzone1.

```
global$ pfbash zonecfg -z kzone1
zonecfg:kzone1> add device
zonecfg:kzone1:device> set storage=dev:/dev/dsk/c9t0d0
zonecfg:kzone1:device> set bootpri=4
zonecfg:kzone1:device> end
```

Example 2-7 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$ pfbash zonecfg -z kzone1
zonecfg:kzone1> select device id=0
zonecfg:kzone1:device> set storage=iscsi://zfssa/
luname.naa.600144F0DBF8AF19000052E820D60003
zonecfg:kzone1:device> end
zonecfg:kzone1> info device
device:
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 Resource Types and Global Properties in *Oracle Solaris Zones Configuration Resources* for more information about these two resource types.

Note:

Best practice is to use an `anet` resource with kernel zones.

Kernel zones must be exclusive-IP zones. For information about exclusive-IP zones, see Networking in Exclusive-IP Non-Global Zones in *Creating and Using Oracle Solaris Zones*.

You can supply additional MAC addresses to support running native (`solaris`) zones inside a kernel zone. See Managing Non-Global Zones in Kernel Zones for more information.

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.

Example 2-8 Adding Network Devices to a Kernel Zone

This example shows how to add a network device to the kernel zone `kzone1`. The ID of 3 specifies the order in which the new `anet` interface is presented to the kernel zone. After booting the zone, the `dladm show-phys -i` command shows information about implicitly created physical links in the kernel zone. The value in the ID column matches the ID that you set with `zonecfg`.

```
global$ pfbash zonecfg -z kzone1
zonecfg:kzone1> add anet
zonecfg:kzone1:anet> set id=3
zonecfg:kzone1:anet> info
anet 1:
id: 3
zonecfg:kzone1:anet> end ; exit
```

```
global$ zoneadm -z kzone1 boot
global$ zlogin kzone1 dladm show-phys -i
LINK            MEDIA        ID        DEVICE       ACTIVE      STANDBY
net0            Ethernet     anet:0    vnic1000     --          zvnet0
net1            Ethernet     anet:3    vnic1001     --          zvnet1
```

Example 2-9 Removing Network Devices From a Kernel Zone

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

```
global$ pfbash zonecfg -z kzone1 info anet
anet:
        configure-allowed-address: true
        id: 0
anet:
        configure-allowed-address: true
        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 `iov` property of the `zonecfg` `anet` resource type. 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 bare metal performance.

SR-IOV must be enabled on the datalink in the global zone in order to enable it on the `anet` resource in a kernel zone. For information about 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 11.4*.

The `iov` property is only supported on kernel zones and native (`solaris`) zones.

When you enable the `iov` property, the ability to suspend and resume the kernel zone and migrate it using warm or live migration is limited to host systems and zones running Oracle Solaris 11.4. See About Migration of Kernel Zones with SR-IOV-Enabled anet Resources for more information.

See Zone Global Properties in *Oracle Solaris Zones Configuration Resources* for information about how to enable and configure the `iov` property of the `anet` resource type.

Tip:

When using some Intel network adapters that support SR-IOV, a virtual function might be the target of malicious behavior. Unexpected software-generated frames can slow traffic between the host system and the virtual switch, which might negatively affect performance. You can work around this issue by configuring all SR-IOV-enabled ports to use VLAN tagging to drop unexpected and potentially malicious frames, See Configuring SR-IOV and VLAN Tagging on an anet Resource for an example.

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

1. Become a zone administrator.

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Enable iov on an anet.

   Using `zonecfg`, set the `iov` property on a selected `anet` resource.

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

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

   ```
   global$ pfbash 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 ; exit
   ```

3. Confirm that the iov property is set for the anet resource in the kernel zone configuration.

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

   For example, on the system `global` and the `anet` resource with ID `0` of the kernel zone `kzone1`:

   ```
   $ zonecfg -z kzone1 info anet id=0
   anet:
           lower-link: net1
           configure-allowed-address: true
           iov: auto
           id: 0
   ```

4. 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 -i
```

The output from this command varies depending on which version of Oracle Solaris is running in the global zone of the host system and in the kernel zone. The following is sample output for selected Oracle Solaris version combinations.

- The global zone and the kernel zone are both running Oracle Solaris 11.4:

```
global$ pfexec zlogin kzone
kzone$ dladm show-phys -i
LINK              MEDIA          ID        DEVICE       ACTIVE      STANDBY
net0              Ethernet       anet:0    vnic1000     ixgbevf0    zvnet0
```

- The global zone is running Oracle Solaris 11.4 and the kernel zone is running Oracle Solaris 11.3:

```
global$ pfexec zlogin kzone
kzone$ dladm show-phys -i
LINK         MEDIA        STATE     SPEED    DUPLEX      DEVICE
net0         Ethernet     down      0        unknown     ixgbevf0
```

- The global zone is running Oracle Solaris 11.3 and the kernel zone is running Oracle Solaris 11.4:

```
global$ pfexec zlogin kzone
kzone$ dladm show-phys -i
LINK              MEDIA          ID        DEVICE       ACTIVE      STANDBY
net0              Ethernet       anet:0    vnic1000     ixgbevf0    --
```

Example 2-10   Confirming the `zonecfg` `iov` Value on an `anet`

The following example shows the `iov` value on `anet 0`. The value is set to `auto`. If set to the default value `off`, it would not be displayed.

```
global$ pfbash zonecfg -z kzone1
zonecfg:kzone1> select anet id=0
zonecfg:kzone1:anet> info
anet:
        lower-link: net1
        configure-allowed-address: true
        iov: auto
        id: 0
```

Example 2-11   Configuring SR-IOV and VLAN Tagging on an `anet` Resource

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

```
global$ pfbash 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 ; exit
```

For more information about setting VLAN IDs and VLAN tagging, see Configuring Virtual LANs in Kernel Zones.

About Migration of Kernel Zones with SR-IOV-Enabled `anet` Resources

For a kernel zone that is using SR-IOV, the ability to suspend and resume the kernel zone and migrate using warm or live migration is limited to host systems and zones running Oracle

Solaris 11.4. If the kernel zone configuration includes the settings `iov=auto` or `iov=on`, migration fails if the source host, target host, or the kernel zone is running an older release.

If you must migrate a kernel zone that has `iov` enabled, and either the kernel zone or the global zone on the source host or target host is running a release that is older than Oracle Solaris 11.4, you must perform a cold migration.

For more information about migration and the limitations of the `iov` property, review:

- Migrating an Oracle Solaris Kernel Zone
- Zone Global Properties in *Oracle Solaris Zones Configuration Resources*

Configuring Network High Availability for SR-IOV-Enabled Kernel Zones

You can achieve network high availability for SR-IOV devices within a kernel zone if the global zone is using datalink multipathing (DLMP) with SR-IOV.

To enable high availability, you must create a DLMP link aggregation of links with SR-IOV enabled in the global zone, then add an `anet` resource in the kernel zone that uses the DLMP aggregation as its lower link. Set the `anet` resource's `iov` property to `auto` and boot the zone.

For more information, see Datalink Multipathing Aggregations in *Managing Network Datalinks in Oracle Solaris 11.4*.

Example 2-12   Configure a DLMP Link Aggregation for Network High Availability in a Kernel Zone

```
global$ pfbash dladm set-linkprop -p iov=on net3
global$ dladm set-linkprop -p iov=on net4
global$ dladm create-aggr -l net3 -l net4 -m dlmp dlmp0
global$ zonecfg -z kz1
zonecfg:kz1> create -t SYSsolaris-kz
zonecfg:kz1> add anet
zonecfg:kz1:anet> set lower-link=dlmp0
zonecfg:kz1:anet> set iov=auto
zonecfg:kz1:anet> set id=0
zonecfg:kz1:anet> end

global$ zoneadm -z kz1 boot
global$ dladm show-aggr -C dlmp0
LINK        PORT           SPEED DUPLEX   STATE     CLIENTS
dlmp0       --             10000Mb full   up        --
            net3           10000Mb full   up        kz1/net0
            net4           10000Mb full   up        dlmp0

global$ zlogin kz1 dladm show-phys -i
LINK              MEDIA          ID       DEVICE      ACTIVE     STANDBY
net0              Ethernet       anet:0   vnic1000    ixgbevf0   zvnet0
```

Using Virtual Functions and Shadow VNICs With 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 system. 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
```

VF can be allocated from a DLMP aggregation. You can set `iov=auto` on a DLMP aggregation, which causes VF to be allocated when there's an available VF resource. An example is shown in Configure a DLMP Link Aggregation for Network High Availability in a Kernel Zone.

Setting `iov=on` over either DLMP or trunk aggregation is prohibited.

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`(8) and `zonecfg`(8) man pages for additional information.

For additional information about VNICs and network configuration, consult *Managing Network Virtualization and Network Resources in Oracle Solaris 11.4*.

Configuring the `suspend` Resource Type

Suspend and resume are supported for a kernel zone only if a kernel zone has a `suspend` resource in its configuration. You must add a `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. The storage device for the suspended image must be large enough to accommodate the amount of memory allocated to the kernel zone in its `capped-memory:physical` property.

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` global property to enable a kernel zone to be suspended automatically instead of shut down when the global zone is shut down. For more information, see autoshutdown Global Property in *Oracle Solaris Zones Configuration Resources*.

Note:

If a kernel zone is using SR-IOV, the ability to suspend and warm or live migrate the kernel zone is limited to host systems and zones running Oracle Solaris 11.4. See About Migration of Kernel Zones with SR-IOV-Enabled anet Resources for more information.

For a more extensive example and background, see *Migratory Solaris Kernel Zones*.

Example 2-13 Configuring a `suspend` Resource to Enable Pausing a Kernel Zone

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

```
global$ pfbash 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
```

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

```
global$ zoneadm -z kz1 suspend
```

Example 2-14 Configure the `suspend` Resource to Enable Warm Migration

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

```
global$ pfbash 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 ; exit
```

See Using Warm Migration to Migrate a Kernel Zone for more information.

See the `solaris-kz`(7) man page for more information about `suspend` resource type 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` resource type to enable and to configure verified boot on a kernel zone. Verified boot and the `verified-boot` resource type are supported only for `solaris-kz` brand zones. For examples and information about properties, see verified-boot Resource Type in *Oracle Solaris Zones Configuration Resources*.

For additional information about certificate verification and verified boot, see the `elfsign(1)` man page and Using Verified Boot in *Securing Systems and Attached Devices in Oracle Solaris 11.4*.

Working with IPoIB and Kernel Zones

You can configure a kernel zone to support InfiniBand (IPoIB) devices by setting properties of the `anet` resource type. Consult Zone Global 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 11.4*.

Configuring Virtual LANs in Kernel Zones

By using Ethernet-based `anet` resources, you can create VNICs inside a kernel zone and configure them to be in their own virtual LAN (VLAN).

- Use a `vlan` resource to add extra VLAN IDs (VIDs) to an existing `anet` resource to create new VLANs. See Configurable Resource Types and Global Properties in *Oracle Solaris Zones Configuration Resources* for more information about the `anet` and `vlan` resource types.
- The `vlan` resource type makes a kernel zone VLAN-aware. The host system forwards to the kernel zone the packets that are meant for these VLANs without stripping the VLAN tag. The kernel zone will then forward the packet to the correct 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:

You are 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 untagged packets that match the zone's MAC addresses are passed on to the zone from the host.

Example 2-15 Configuring a Kernel Zone with Additional VLANs

Configure a zone `kzone0` with a `mac-address` of `0:1:2:3:4:5`, PVID of `11`, and two additional VIDs of `45` and `46`.

```
global$ pfbash zonecfg -z kzone0
zonecfg:kzone0> create -t SYSsolaris-kz
zonecfg:kzone0> select anet id=0
zonecfg:kzone0> set mac-address=0:1:2:3:4:5
zonecfg:kzone0:anet> set vlan-id=11
zonecfg:kzone0:anet> add vlan
zonecfg:kzone0:anet:vlan> set vlan-id=45
zonecfg:kzone0:anet:vlan> end
zonecfg:kzone0:anet> add vlan
zonecfg:kzone0:anet:vlan> set vlan-id=46
zonecfg:kzone0:anet:vlan> end
zonecfg:kzone0:anet> info vlan
    vlan 0:
        vlan-id: 45
    vlan 1:
        vlan-id: 46
zonecfg:kzone0:anet> end
zonecfg:kzone0> commit ; exit
```

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

```
global$ dladm show-vnic
LINK               OVER     SPEED  MACADDRESS     MACADDRTYPE    IDS
kzone0/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 a `0:1:2:3:4:5, 11` tuple have the VID stripped by the system `global` and passed on to the kernel zone `kzone0`, so `kzone0` never sees the VID `11` tag. Frames with `0:1:2:3:4:5, 45` and `0:1:2:3:4:5, 46` will be passed to `kzone0` with their tags VID `45` and `46`.

Inside `kzone0`, if there is a VLAN datalink `vlan45` with VID of 45, the virtual switch in `kzone0` will strip VID `45` from the frame and pass the frame to `vlan45`. All the frames originating from `vlan45` datalink inside `kzone0` will be tagged by the virtual switch in `kzone0` and passed onto the `anet` in the host. The host `anet` will pass the frames directly to the NIC to be sent out.

Example 2-16 Display the List of VLAN IDs Supported in the Kernel Zone

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

```
global$ zlogin kzone0
kzone0$ 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 cases such as a cloud deployment, you might not know ahead of time what values the kernel zone needs to use for MAC addresses and the VLAN IDs of its VNICs. In such cases, you have two configuration options:

- You can specify prefixes of allowed MAC addresses and ranges of allowed VLAN IDs.
- You can enable the kernel zone to create a VNIC with any valid MAC address or VLAN ID.

Tip:

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 `anet` resources.

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

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

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

1. Become a zone administrator.

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

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.

   ```
   $ pfbash 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 allowed-vlan-ids 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 ; exit
   ```

4. Boot the kernel zone.

   ```
   $ zoneadm -z kernel-zone boot
   ```

5. Log in to the kernel zone.

   ```
   $ zlogin kernel-zone
   ```

6. Verify in the kernel zone the allowed addresses and IDs.

   To determine which MAC prefixes and VLAN IDs are allowed, use the `dladm show-phys` command with the `-o` option to specify output fields. For example, to verify for `kzone1`:

   ```
   global$ zlogin kzone1
   kzone1$ dladm show-phys -o link,media,id,allowed-addresses,allowed-vids
   LINK            MEDIA         ID        ALLOWED-ADDRESSES ALLOWED-VIDS
   net0            Ethernet      anet:0    fa:16:3f,         100-199,
                                           fa:80:20:21:22    400-498,500
   ```

3

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:

- Installing a Kernel Zone
- Uninstalling a Kernel Zone
- Shutting Down, Halting, and Rebooting a Kernel Zone
- Cloning a Kernel Zone

For general information about zone installation and zone cloning concepts, see How Non-Global Zones Are Created in *Introduction to Oracle Solaris Zones*.

Installing a Kernel Zone

Before you can install a kernel zone, you must configure it as described in Configuring Oracle Solaris Kernel Zones. After configuration, you can choose the appropriate zone installation method:

- Create one kernel zone that is identical to the global zone configuration – Directly Installing a Kernel Zone
- Create multiple kernel zones with identical requirements – Using an AI Manifest to Install a Kernel Zone
- Create multiple kernel zones with identical requirements – Using a sysconfig Profile to Install a Kernel Zone
- Duplicate an existing zone, or duplicate and modify an existing zone – Using an Installation Image to Install a Kernel Zone
- Copy an existing installed zone from your system and add the copy to your system – Cloning a Kernel Zone

During zone installation, a log is recorded in two locations. The contents of the logs are the same.

- In the global zone in the `/var/log/zones/` directory
- In the installed zone in the `/zones/`*`zonename`*`/root/var/log/zones/` directory

Directly Installing a Kernel Zone

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

In a direct installation, the installer runs in the global zone. 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 installs the exact version of Oracle Solaris that is running in the global zone. To install a different version of Oracle Solaris, you must choose a different installation method.

How to Directly Install a Kernel Zone

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Install the kernel zone.

   ```
   global$ pfbash zoneadm -z kzone 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 Troubleshooting Local Package Repositories in *Creating Package Repositories in Oracle Solaris 11.4* for more information.

3. Review the installation logs.

   For their location, see Installing a Kernel Zone.

4. Boot the kernel zone.

   ```
   global$ zoneadm -z kzone boot
   ```

   You can now log in to the zone. Oracle Solaris provides several login methods. The methods for kernel zones are identical to the methods for non-global zones. What you are going to do in the zone determines your login method. See How to Create and Deploy a Non-Global Zone in *Creating and Using Oracle Solaris Zones*.

Example 3-1 Installing a Kernel Zone Using Direct Installation

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

```
global$  pfbash 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 an AI Manifest to Install a Kernel Zone

You can use an Automated Installation (AI) manifest when you need to install multiple kernel zones with specific resource and package configurations that are different from those in the global zone.

Observe the following requirements and guidelines when installing an alternate AI manifest 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.

- 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 that you are installing. See Using an Installation Image to Install a Kernel Zone for an example.

How to Install a Kernel Zone by Using an AI Manifest

Review the requirements and guidelines in Using an AI Manifest to Install a Kernel Zone.

This procedure provides the `zoneadm` command to install a kernel zone with an AI manifest and detailed examples.

For additional information about developing and customizing an AI manifest, see Chapter 5, Specifying Criteria for AI Manifests and System Configuration Profiles in *Customizing Automated Installations With Manifests and Profiles*.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Install the kernel zone by using a specified AI manifest.

   ```
   global$ zoneadm -z kzone install -m path-to-manifest
   ```

3. Review the installation logs.

   For their location, see Installing a Kernel Zone.

Example 3-2 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$ pfbash 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
```

Example 3-3 Installing a Kernel Zone Using an 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 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 IDs `0` and `1` exist in the `kzone1` zone configuration, you can use the following command to install in the kernel zone `kzone1`:

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

Using a `sysconfig` Profile to Install a Kernel Zone

You can use `sysconfig` profile when you need to install multiple kernel zones with specific resource and package configurations that are different from those in the global zone.

How to Install a Kernel Zone by Using a `sysconfig` Profile

Create the `sysconfig` profile and put it in a location that is accessible to the system you use to install the kernel zone. For more information, see Chapter 3, Working With System Configuration Profiles in *Customizing Automated Installations With Manifests and Profiles* and the `sysconfig`(8) and `solaris-kz`(7) man pages.

For more information about using a `sysconfig` profile to install kernel zones, see the `zoneadm`(8) man page.

This procedure provides a detailed example of how to use the `zoneadm` command to install a kernel zone with a `sysconfig` profile.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Install the kernel zone by using a sysconfig profile.

   ```
   global$ pfbash zoneadm -z kzone install -c sysconfig-profile
   ```

3. Review the installation logs.

   For their location, see Installing a Kernel Zone.

Example 3-4 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$ pfbash 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
```

Using an Installation Image to Install a Kernel Zone

You can install a kernel zone from an installation image alone or combined with an AI manifest with specific resource and package configurations.

Observe the following requirements and guidelines when installing a kernel zone from an installation image:

- The version of Oracle Solaris in the installation image must support kernel zones, so it must be at least Oracle Solaris 11.2. See Verifying Hardware and Software Support on Kernel Zone Hosts.

- You must specify the complete path to the ISO image.
- Interactive text installations and automated installations from media are both supported. Live Media installation is not supported for kernel zones. For more information, see:
  - Chapter 3, Using the Text Installer in *Manually Installing an Oracle Solaris 11.4 System*
  - Chapter 8, Automated Installations That Boot From Media in *Automatically Installing Oracle Solaris 11.4 Systems*

This section provides the following procedures for installing a kernel zone from an installation image:

- How to Install a Kernel Zone From an Installation Image
- How to Install a Kernel Zone From an Installation Image and an AI Manifest

How to Install a Kernel Zone From an Installation Image

- Review the requirements and guidelines in Using an Installation Image to Install a Kernel Zone.
- Ensure that the installation image is accessible to the system you use to install the kernel zone.

During an Oracle Solaris installation from an ISO file, the kernel zone is booted and you are connected to the zone console. For how to use the zone console, see How to Create and Deploy a Non-Global Zone 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.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.
2. Install the kernel zone by using the Oracle Solaris installation image.

   ```
   global$ pfbash zoneadm -z kzone install -b path-to-iso-file
   ```
3. Review the installation logs.

   For their location, see Installing a Kernel Zone.

Example 3-5 Installing an Kernel Zone From an Installation Image

This example installs the image located at `/var/tmp/solaris-media.iso` to the kernel zone, `kzone2`.

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

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

- Review the requirements and guidelines in Using an Installation Image to Install a Kernel Zone.

- Ensure that the installation image and AI manifest are accessible to the system you use to install the kernel zone.

If you do not need to provide specific resource and package configuration information, go instead to How to Install a Kernel Zone From an Installation Image.

1. Become a zone administrator.

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Install the kernel zone by using an installation image ISO file and an AI manifest.

   ```
   global$ pfbash zoneadm -z kzone install -b path-to-iso-file -m path-to-manifest
   ```

   For more information about the `zoneadm install` options, see the `zoneadm`(8) man page.

3. Review the installation logs.

   For their location, see Installing a Kernel Zone.

Example 3-6 Installing a Kernel Zone From an Installation Image and Using an AI Manifest

This example installs the image located at `/var/tmp/solaris-media.iso` to the kernel zone `kzone2` and uses the AI manifest `/var/tmp/kz_manifest.xml` to provide specific resource and package configurations:

```
global$ pfbash 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. For more information, see How to Uninstall and Remove a Zone in *Creating and Using Oracle Solaris Zones*.

Shutting Down, Halting, and Rebooting a Kernel Zone

Use the `zoneadm shutdown`, `zoneadm halt`, and `zoneadm reboot` commands to shut down, halt, and reboot a kernel zone.

Tip:

If you want a zone to boot automatically when the host system reboots, set the `autoboot` property to `true`.

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 exist in each clone. Cloning a zone is an efficient way to add additional zones that have similar requirements.

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 Cloning a Kernel Zone by Using the zoneadm clone Command.
- Use a Unified Archive file if you need to clone multiple zones for a large deployment. See Cloning and Deploying a Kernel Zone by Using a Unified Archive.

  For more information, see Chapter 2, Working With Unified Archives in *Using Unified Archives for System Recovery and Cloning in Oracle Solaris 11.4*.

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

Example 3-7 Cloning a Kernel Zone by Using the `zoneadm clone` Command.

The following example demonstrates how to clone the kernel zone `kzone1` to the kernel zone `kzone2` on the host system `global`.

```
global$ pfbash 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 3-8 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, see *Using Unified Archives for System Recovery and Cloning in Oracle Solaris 11.4*.

```
global$ pfbash 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
```

4

Live Zone Reconfiguration of Kernel Zones

Use Live Zone Reconfiguration to reconfigure or report on the live configuration of kernel zones while the zones are running.

To view a table that shows Live Zone Reconfiguration support for resources and properties in `solaris` zones and kernel zones, see Live Zone Reconfiguration Support or Restriction for Resource Types and Global Properties in *Oracle Solaris Zones Configuration Resources*.

Live Zone Reconfiguration of Kernel Zones

Use Live Zone Reconfiguration to reconfigure or report on the live configuration of kernel zones while the zones are running.

This section covers the following information:

- zonecfg Utility Edit Modes
- About Live Zone Reconfiguration of Kernel Zones
- Temporary Changes to the Live Zone Configuration
- Persistent Changes to the Live Zone Configuration
- Live Zone Reconfiguration Dry Run
- Reloading a Live Zone Configuration

`zonecfg` Utility Edit Modes

Use the `zonecfg` utility edit modes to make configuration changes to the zone. You can change either the persistent stored configuration or the running live configuration. The `zonecfg` utility supports the following edit modes for use with Live Reconfiguration:

Default mode
Create, modify, and list the persistent zone configuration stored on the stable storage. Parameters you changed in the default mode do not affect a running zone at the time you make the changes. The default mode is the primary way to maintain the zone configuration. This mode is backwards compatible.
To have the changes made in default mode take effect in the running zone, you must issue one of the following `zoneadm` commands:

- Use the `zoneadm apply` command to load the updated persistent zone configuration so it is applied to the running zone.
- Use the `zoneadm reboot` command to reboot the zone and read the updated persistent zone configuration.

Live mode
Retrieve, inspect and edit the running live zone configuration. The live mode is available for a running zone only. Parameters you change in live mode take effect immediately after you use the `commit` subcommand to enable them in the live zone configuration.
Changes made in live mode are temporary. The changes remain active until the next zone reboot. For more information, see Temporary Changes to the Live Zone Configuration.
To make the modified live zone configuration permanent, make the same changes to the permanent configuration. To change both the live configuration and the permanent configuration, first edit the permanent configuration and then use the `zoneadm apply` command to apply the changes.
To enable live mode, use the -r option with the `zonecfg` command, which retrieves the live zone configuration instead of the persistent zone configuration.

```
global$ pfexec zonecfg -z kzone -r
```

You can work with the `zonecfg -r` command just as you do in default mode. The full set of `zonecfg` subcommands and both the interactive and the batch mode are supported.
Not all resources can be reconfigured in the live configuration. For a list of supported resource types and properties, see Live Zone Reconfiguration Support or Restriction for Resource Types and Global Properties in *Oracle Solaris Zones Configuration Resources*.

About Live Zone Reconfiguration of Kernel Zones

Use Live Zone Reconfiguration to perform the following tasks on a running kernel zone:

- Report on and inspect the current live zone configuration
- Make temporary changes in the live zone configuration
- Apply changes made in the persistent configuration to the live zone configuration

Use the `zonecfg` and `zoneadm` commands to administer Live Zone Reconfiguration. You can make temporary or persistent changes to the zone configuration. Temporary changes are active until the next reboot. You do not need to reboot for changes to be applied to the persistent configuration.

Services are available within the zone with no downtime when you make the following configuration changes:

- Changing resource controls
- Changing network configuration
- Changing the CPU resource pool
- Changing the memory size
- Adding or removing file systems
- Adding or removing virtual and physical devices

For a list of supported resource types and properties during live zone reconfiguration, see Live Zone Reconfiguration Support or Restriction for Resource Types and Global Properties in *Oracle Solaris Zones Configuration Resources*.

Temporary Changes to the Live Zone Configuration

You might want to make only temporary changes in the configuration of a running zone. For example, you might want to remove a resource from a zone temporarily while maintenance is performed on a device, or allocate a resource temporarily for a special purpose but not have it be present for the entire run of the zone. Such removals or additions of resources should not be done in the persistent zone configuration because they would cause a failure when the zone reboots and the resource is no longer available.

Parameters changed temporarily in live mode take effect immediately after you issue the `zonecfg commit` command. These changes are valid until the next zone reboot.

Persistent Changes to the Live Zone Configuration

You use the `zoneadm apply` command to apply changes from the persistent zone configuration to the live zone configuration. You do not have to reboot for the changes to affect the running zone. For an example, see How to Make Persistent Configuration Changes to a Live Zone.

Live Zone Reconfiguration Dry Run

You can test run the effects of changes to the live zone configuration before putting those changes into effect by using the following options to the `zonecfg commit` and `zoneadm apply` commands:

-n
Dry run mode. The command shows the effects of the changes to the configuration, but applies no changes to the running zone. Use the dry run mode to preview the actions that would be performed if you issued the `zonecfg commit` or `zoneadm apply` command to impact the live zone configuration.

-q
Quiet mode. This mode suppresses all system messages and returns a status code only.

Reloading a Live Zone Configuration

If the configuration of a running zone changes externally while you are modifying the configuration in either default mode or live mode, the `zonecfg commit` command will return an error. Some scenarios where this might occur include another administrator modifying the configuration, modifying resource controls, or changing network parameters of the zone using network administration commands.

If the configuration of a running zone changes during live reconfiguration, use the `zonecfg reload` subcommand to load the external configuration changes:

- If you issue the `zonecfg reload` command in default mode, the command discards any uncommitted changes you have made and reloads the configuration from persistent storage.
- If you issue the `zonecfg reload` command in live mode (the -r option), the command discards any uncommitted changes and retrieves an up-to-date live configuration of the running zone.

After the configuration is reloaded, you can repeat the configuration changes and commit.

See How to Recover From a Failure While Committing Live Zone Configuration Changes for an instructions to reload a zone configuration.

Performing a Live Zone Reconfiguration

This section provides the following procedures to perform common live zone reconfiguration tasks:

- How to Inspect the Live Configuration of a Running Zone
- How to Preview the Effect of a Live Zone Configuration
- How to Make Persistent Configuration Changes to a Live Zone
- How to Make Temporary Changes to the Running Zone
- How to Recover From a Failure While Committing Live Zone Configuration Changes

How to Inspect the Live Configuration of a Running Zone

Perform this procedure to view and export the configuration of a running zone.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Display information about the live zone configuration.

   ```
   global$ pfbash zonecfg -z kzone -r info
   ```

3. Export the live configuration.

   ```
   global$ zonecfg -z kzone -r export -f exported.cfg
   ```

How to Preview the Effect of a Live Zone Configuration

Review the following:

- Live Zone Reconfiguration Dry Run
- Appendix A, Resource Types and Global Properties That Support Live Zone Reconfiguration in *Oracle Solaris Zones Configuration Resources*

Perform this procedure to review the live zone configuration changes that would be made, before you make a final commitment of those changes.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Start the zonecfg utility in live mode and configure the desired zone changes.

   ```
   global$ pfbash zonecfg -z kzone -r
   zonecfg:kzone> Make zone configuration changes
   ```

3. View the actions that would be performed by the reconfiguration.

   The -n option prevents actual commitment of the zone changes.

```
zonecfg:kzone> commit -n
```

To make the previewed changes to the live zone configuration, issue the `zonecfg commit` command without using the -n option.

How to Make Persistent Configuration Changes to a Live Zone

Review the following documentation:

- zonecfg Utility Edit Modes
- Persistent Changes to the Live Zone Configuration
- Appendix A, Resource Types and Global Properties That Support Live Zone Reconfiguration in *Oracle Solaris Zones Configuration Resources*

Perform this procedure to make live zone configuration changes that persist across reboots of the zone.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Make changes to the zone in default mode.

   ```
   global$ pfbash zonecfg -z kzone "zonecfg-commands"
   ```

   Specify one or more `zonecfg` subcommands.

   ```
   kzone> "zonecfg-commands"
   ```

   For the format, see the `zonecfg`(8) man page.

3. Apply the changes to the live configuration.

   ```
   global$ pfbash zoneadm -z kzone apply
   ```

Example 4-1 Reducing the Number of Virtual CPUs in the Live Zone Configuration

This example shows a running kernel zone `kz1` that has 16 virtual CPUs configured. The persistent configuration is changed to set the number of VCPUs to 8 and applied to the live configuration. The output shows what happens if the kernel zone cannot stop using some of the CPUs. This might occur if the kernel zone contains a `solaris` zone that is configured with the `dedicated-cpu` resource type, for instance.

The live zone reconfiguration tries to satisfy the request by skipping those CPUs that cannot be removed from the kernel zone (the guest) while trying to remove others. When the specified number of CPUs cannot be removed, the operation succeeds partially and the output shows the new number of virtual CPUs.

```
global$ pfbash zonecfg -z kz1 -r info virtual-cpu
virtual-cpu:
        ncpus: 16
global$ zonecfg -z kz1 'select virtual-cpu;set ncpus=8;end'
global$ zoneadm -z kz1 apply
zone 'kz1': Checking: Modifying virtual-cpu ncpus=8
zone 'kz1': Applying the changes
zone 'kz1': error: dr-cpu failed for cpu id=15: Operation was blocked
zone 'kz1': error:        status: CPU is configured for use by the guest
zone 'kz1': error: dr-cpu failed for cpu id=14: Operation was blocked
zone 'kz1': error:        status: CPU is configured for use by the guest
```

```
...
operation continues to try to remove 8 virtual CPUs
...
zone 'kz1': warning: operation succeeded partially for virtual cpus (requested:
8, final: 12)

global$ zonecfg -z kz1 -r info virtual-cpu
virtual-cpu:
        ncpus: 12
```

How to Make Temporary Changes to the Running Zone

Review the following documentation:

- zonecfg Utility Edit Modes
- Temporary Changes to the Live Zone Configuration
- Live Zone Reconfiguration Support or Restriction for Resource Types and Global Properties in *Oracle Solaris Zones Configuration Resources*

Perform this procedure to temporarily change the live configuration of a running zone and then restore the persistent configuration to undo the change.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Change the zone configuration.

   The sample command adds a disk in live mode and shows the command output.

   > **Note:**
   >
   > The `zonecfg commit` command is not required. The `zonecfg` utility commits the changes when the command exits.

   ```
   global$ pfbash zonecfg -z kzone -r 'add device;set storage=dev:/dev/dsk/
   cNtXd;end'
   zone 'kzone': Checking: Adding device storage=dev:/dev/dsk/cNtXd
   zone 'kzone': Applying the changes
   ```

3. When you no longer need the configuration change, return the zone to the persistent zone configuration.

   The sample command removes the temporary configuration change.

   ```
   global$ zoneadm -z kzone apply
   zone 'kzone': Checking: Removing device storage=dev:/dev/dsk/cNtXd
   zone 'kzone': Applying the changes
   ```

   Alternatively, you can reboot the zone to discard the live zone configuration changes and return to the persistent zone configuration.

If the `commit` operation reports an error, see How to Recover From a Failure While Committing Live Zone Configuration Changes.

How to Recover From a Failure While Committing Live Zone Configuration Changes

The configuration of a running zone can change externally while a live zone configuration is being edited. When this conflict occurs, the `zonecfg commit` command returns an error.

Perform this procedure to correct the error by reloading the zone configuration to show the updated version and then making your edits again.

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

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Issue the reload subcommand, make the correct configuration changes, and commit the change.

   This step assumes that you are still in the `zonecfg` session that failed to commit your temporary configuration changes.

   ```
   zonecfg:kzone> reload
   zonecfg:kzone> temporary-configuration-changes
   zonecfg:kzone> commit
   ```

Example 4-2 Recovering From Failed Temporary Zone Configuration Changes

The following example shows temporary configuration changes, an error message that the changes failed, recovery steps, and output confirming that the changes now succeeded.

```
Example configuration change in live mode
global$ pfbash zonecfg -z kzone1 -r
zonecfg:kzone1> add anet;set lower-link=net1;set maxbw=2G;end
zonecfg:kzone1> commit
zone 'kzone1': error: the live configuration has changed externally.
Trying to commit changes to externally changed live configuration
Please use reload to start again. Your local changes will be lost.

Reload the configuration
zonecfg:kzone1> reload
Are you sure you want to reload (y/[n])? y
Repeat the configuration changes you previously attempted
zonecfg:kzone1> add anet;set lower-link=net1;set maxbw=2G;end
zonecfg:kzone1> commit
Command output shows the configuration changes now succeed
zone 'kzone1': Checking: Adding anet id=2
zone 'kzone1': Applying the changes
…
```

Memory Live Zone Reconfiguration for Kernel Zones

Memory live zone reconfiguration extends the live zone reconfiguration feature to include the dynamic resizing of a running kernel zone's memory size.

Prior to the introduction of this feature, you could change the memory size of a kernel zone only when booting or rebooting that kernel zone.

Note:

This feature applies only to `solaris-kz` branded zones (kernel zones).

To use this feature, you must perform the following tasks:

- Enable memory live zone reconfiguration by setting the `host-compatible` property
- Configure memory allocation for the kernel zone in one of the following ways:
  - Specify a page size policy and page size by setting the `capped-memory:pagesize-policy` and `capped-memory:pagesize` properties, respectively
  - Reserve memory for the kernel zone's memory reservation pool by setting the `capped-memory:memory-reserve` property

The `capped-memory:pagesize-policy` and `capped-memory:memory-reserve` properties are mutually exclusive. So, when you set one of the property's value, ensure that the other property's value is cleared.

This section covers the following topics:

- Memory Live Zone Reconfiguration Software Requirements
- Enabling the Memory Live Zone Reconfiguration Feature
- Configuring Memory Live Zone Reconfiguration
- Modifying the Amount of Kernel Zone Physical Memory
- Memory Live Zone Reconfiguration Failure Examples

Memory Live Zone Reconfiguration Software Requirements

To use the memory live zone reconfiguration feature, ensure that your Oracle Solaris host system runs at least the following Oracle Solaris 11.4 Support Repository Update (SRU):

To use this feature, you must perform the following tasks:

- **SPARC:** Oracle Solaris 11.4 SRU 45
- **x86:** Oracle Solaris 11.4 SRU 51

In addition, ensure that your Oracle Solaris guest system runs at least the following Oracle Solaris 11 SRU:

- **SPARC:** Oracle Solaris 11.3 SRU 8
- **x86:** Oracle Solaris 11.4 SRU 51

Enabling the Memory Live Zone Reconfguration Feature

To enable memory live zone reconfiguration on a kernel zone, use the `zonecfg` command to set the `host-compatible` property to one of the following values:

- `native`
- `level2`

- `level1,memlzr`
- `memlzr`

Note that the `native`, `level2`, and `level1,memlzr` values have the same behavior. For descriptions of these values, see the `solaris-kz`(7) man page.

After you use the `zonecfg` command to update a kernel zone's configuration, you must reboot the kernel zone for the changes to take effect.

Note:

The memory live zone reconfiguration feature introduces an incompatible memory layout for a kernel zone guest. This memory layout might affect live or warm migrations of such a kernel zone guest to a host that does not support the feature. As a result, only enable this feature by default for an existing kernel zone that you intend to migrate to a host that supports the feature. See Kernel Zone Migration Failures.

New kernel zones have memory live zone reconfiguration enabled by default. Both the `SYSsolaris-kz` and `SYSsolaris-kz-minimal` templates include the `host-compatible=memlzr` setting.

To enable memory live zone reconfiguration on an existing kernel zone, modify its `host-compatible` property value manually.

How to Enable Memory Live Zone Reconfiguration

Perform this procedure to add two automatically generated MAC addresses to a kernel zone.

1. Ensure that your Oracle Solaris host system and your Oracle Solaris guest systems run the required versions of Oracle Solaris 11 operating system.

   See Memory Live Zone Reconfiguration Software Requirements.

2. Enable the memory live zone reconfiguration feature.

   a. Configure the kernel zone.

   ```
   # zonecfg -z kz-name
   ```

   b. Set the `host-compatible` property to a value that supports memory live zone reconfiguration.

   Valid property values are `native`, `memlzr`, `level2`, and `level1,memlzr`.

   ```
   zonecfg:kz-name> set host-compatible=mlzr-value
   ```

   c. Commit the change.

   ```
   zonecfg:kz-name> commit
   ```

   d. Exit `zonecfg`.

   ```
   zonecfg:kz-name> exit
   ```

   e. Reboot the kernel zone.

   ```
   # zoneadm -z kz-name reboot
   ```

The following example commands enable memory live zone reconfiguration on an SPARC host system. The commands set the `host-compatible` property value to `level2` for the `somekz` kernel zone, commit the change, and reboot the kernel zone:

```
# zonecfg -z somekz
zonecfg:somekz> set host-compatible=level2
zonecfg:somekz> commit
zonecfg:somekz> exit
# zoneadm -z somekz reboot
```

Configuring Memory Live Zone Reconfiguration

You can configure memory live zone reconfiguration for a kernel zone in one of the following ways:

- `zonecfg` **method:** First, specify any valid value of the `capped-memory:pagesize-policy` property to use the memory live zone reconfiguration feature. For more information, see About Memory Size Page Policy and Physical Memory.
- **Memory Reservation Pool (MRP) method:** Configure a kernel zone to use an MRP for allocating its memory. When a kernel zone uses an MRP, memory is allocated from and returned to the pool, as appropriate. Note that a request to add memory fails if the pool has insufficient memory reserved. See Memory Live Zone Reconfiguration Software Requirements.

  Note that the `svc:/system/memory-reserve:zone` service enables you to perform a live or warm migration of a kernel zone that uses an MRP. Configure this capability by setting `config/pagesize-policy=fixed` and setting the `config/pagesize` property value to a valid page size.

Ensure that you have the memory live zone configuration feature enabled on your kernel zone. See How to Enable Memory Live Zone Reconfiguration.

Note:

You cannot set `capped-memory:pagesize-policy=fixed` unless memory live zone reconfiguration is enabled and the `capped-memory:pagesize` property has no value. See Memory Reservation Pools for Kernel Zones.

This example shows how to use the `zonecfg` command to configure memory live zone reconfiguration for the `somekz` kernel zone.

The following example `pagesize -a` output shows that 256 Mbytes, 2 Gbytes, and 16 Gbytes are valid kernel zone page size values on a SPARC system:

```
sparc-host# pagesize -a
8192
65536
4194304
268435456
2147483648
17179869184
```

Use one of the valid page size values output by the `pagesize -a` command as the value of the `capped-memory:pagesize` property.

The following example commands set the `capped-memory:pagesize-policy` property value to `fixed` for the `somekz` kernel zone, set the memory page size to the minimum SPARC platform value of 256 Mbytes (`256M`), commit the change, and reboot the kernel zone:

```
# zonecfg -z somekz
zonecfg:somekz> set capped-memory:pagesize-policy=fixed
zonecfg:somekz> set capped-memory:pagesize=256M
zonecfg:somekz> commit
zonecfg:somekz> exit
# zoneadm -z somekz reboot
```

Modifying the Amount of Kernel Zone Physical Memory

The memory live zone reconfiguration feature permits you to change the size of a kernel zone's memory. Use the `zonecfg` command to specify the new kernel zone memory size by setting the `capped-memory:physical` property. The new memory size must be a multiple of the page size used by the kernel zone memory. You can obtain the page size of the running kernel zone by using the `kstat2` command.

Note that the maximum value of the `capped-memory:physical` property is 2 TBytes.

If `capped-memory:pagesize-policy=fixed`, the kernel zone guest address space consists of memory pages of a size that you specify as the value of `capped-memory:pagesize`. As a result, you can modify the amount of kernel zone memory only changing the number of pages. For example, if `capped-memory:pagesize=256M`, you can increase or decrease that kernel zone's memory only in 256-Mbyte increments. If `capped-memory:pagesize-policy` is not set to `fixed`, the operating system chooses the page size during the kernel zone boot operation.

An attempt to resize the kernel zone memory can fail, succeed, or partially succeed. A partial success results in the kernel zone memory being resized but not by the requested amount. In all cases, the live value of the `capped-memory:physical` property is updated to represent the current size of the kernel zone memory. See Memory Size Change Warning and Error Messages.

To eliminate a partial success of a memory live zone reconfiguration resize request, configure the strict method by setting the `capped-memory:memlzr` property value to `strict`.

A strict method operation succeeds only when the kernel zone's resized memory exactly matches the requested memory size and the `zoneadm apply` command returns `0`. Otherwise, the `zoneadm apply` command restores the original size of the kernel zone's memory and returns `1`. See Strict Method Memory Resize Request Failure.

Use the `kstat2` command on a running kernel zone to view the page size stored as a statistic in `kstat:/zones/`*`zone-name`*`/memory/usage:pagesize`.

The following example `kstat2` command shows the page size statistic for the running `kz1` kernel zone:

```
# kstat2 kstat:/zones/kz1/memory/usage:pagesize
kstat:/zones/kz1/memory/usage
        pagesize                        268435456 bytes
```

How to Modify the Amount of Kernel Zone Physical Memory

1. Ensure that you have enabled and configured the memory live zone reconfiguration feature.

See How to Enable Memory Live Zone Reconfiguration and Configuring Memory Live Zone Reconfiguration.

2. Change a kernel zone's physical memory size by specifying the new amount of physical memory as the value of the `capped-memory:physical` property.

   a. Configure the kernel zone.

      ```
      # zonecfg -z kz-name
      ```

   b. Specify the value of the `capped-memory:physical` property.

      new-phys-memory-size is an integer with an optional scale unit (`K` for kilobytes, `M` for metabytes, `G` for gigabytes, and `T` for terabytes).

      ```
      zonecfg:kz-name> set capped-memory:physical=new-phys-memory-size
      ```

   c. (Optional) Configure the strict method by setting the `capped-memory:memlzr` property to the `strict` value.

      ```
      zonecfg:kz-name> set capped-memory:memlzr=strict
      ```

   d. Commit the change.

      ```
      zonecfg:kz-name> commit
      ```

   e. Exit `zonecfg`.

      ```
      zonecfg:kz-name> exit
      ```

   f. Apply the configuration changes to the running kernel zone.

      ```
      # zoneadm -z kz-name apply
      ```

   The following example commands set the `capped-memory:physical` property value to 32 Gbytes (`32G`) for the `somekz` kernel zone, commit the change, and apply the configuration changes to the running kernel zone:

   ```
   zonecfg -z somekz
   zonecfg:somekz> set capped-memory:physical=32G
   zonecfg:somekz> commit
   zonecfg:somekz> exit
   # zoneadm -z somekz apply
   ```

Memory Live Zone Reconfiguration Failure Examples

This section includes example warnings and errors that you might see when you use the memory live zone reconfiguration feature.

Examples include the following:

- Kernel Zone Boot Failures
- Kernel Zone Migration Failures
- Memory Live Zone Reconfiguration Configuration Change Failures
- Memory Size Change Warning and Error Messages
- Strict Method Memory Resize Request Failure

Kernel Zone Boot Failures

- The following example shows that the `kz1` kernel zone fails to boot when insufficient pages of the specified page size are available:

```
# zoneadm -z kz1 boot
error: Not enough contiguous free memory pages to allocate
capped-memory:physical value of 16GB for zone.  Consider using
memory-reserve(8s) to facilitate reserving memory for Kernel Zones
```

- The following example shows that the `kz1` kernel zone fails to boot when the specified page size is invalid:

```
# zoneadm -z kz1 boot
error: The capped-memory:pagesize value 10G is not supported on this host.
```

Kernel Zone Migration Failures

- The following example shows that a live migration of the `mykz` kernel zone, which has memory live zone reconfiguration enabled, fails when the `ssh://mem@system1.example.com` target host runs an Oracle Solaris version that does not support the feature:

```
root:sparc-host::~# zoneadm -z mykz migrate ssh://mem@system1.example.com
zoneadm: zone 'mykz': Importing zone configuration on destination.
zoneadm: zone 'mykz': configuration failed:
-----BEGIN CONFIGURATION-----
create -b
set brand=solaris-kz
set host-compatible=memlzr
...
add capped-memory
set physical=12G
set pagesize-policy=fixed
set pagesize=256M
end
...
-----END CONFIGURATION-----
Error received from the target machine:
On line 3:
memlzr: Invalid argument
```

  As expected, the live migration fails even if an existing kernel zone configuration is on the target host that does not support the feature.

- The following example shows that a live migration of the `kz1` kernel zone fails when the `ssh://system2` target host does not support the specified `capped-memory:pagesize` property value:

```
# zoneadm -z kz1 migrate ssh://system2
zoneadm: zone 'kz1': Importing zone configuration on destination.
zoneadm: zone 'kz1': configuration failed:
-----BEGIN CONFIGURATION-----
create -b
set brand=solaris-kz
set cpu-arch=migration-class1
set hostid=0x6f038b0b
add anet
set configure-allowed-address=true
set id=0
end
add device
set storage=iscsi://system2/luname.naa.600144f00010e0767dda6076931b0001
set bootpri=0
set id=0
end
```

```
add capped-memory
set physical=12G
set pagesize-policy=fixed
set pagesize=2G
end
add keysource
set raw="{base64}9/7hHuucihhP6YTwCcjaqg=="
end
-----END CONFIGURATION-----
Error received from the target machine:
On line 16:
"fixed" is not a valid pagesize-policy name
```

Memory Live Zone Reconfiguration Configuration Change Failures

- The following example shows that the `kz1` kernel zone fails to boot when insufficient pages of the specified page size are available:

  ```
  # zoneadm -z kz1 boot
  error: Not enough contiguous free memory pages to allocate
  capped-memory:physical value of 16GB for zone.  Consider using
  memory-reserve(8s) to facilitate reserving memory for Kernel Zones
  ```

- The following example shows that the `kz1` kernel zone fails to boot when the specified page size is invalid:

  ```
  # zoneadm -z kz1 boot
  error: The capped-memory:pagesize value 10G is not supported on this host.
  ```

Memory Size Change Warning and Error Messages

- The following example shows that `zonecfg` does not accept a `capped-memory:pagesize` property value that is smaller than 2 Mbytes for the x86 `kz1` kernel zone:

  ```
  # zonecfg -z kz1
  zonecfg:kz1:capped-memory> set pagesize=1m
  pagesize value is too small. It must be at least 2M.
  zonecfg:kz1:capped-memory> set pagesize=2m
  zonecfg:kz1:capped-memory> end
  zonecfg:kz1> verify
  zonecfg:kz1> exit
  ```

- The following example shows that the MRP service does not accept a `config/pagesize` property value that is smaller than 2 Mbytes on an x86 system:

  ```
  # svcs -xv svc:/system/memory-reserve:zones
  svc:/system/memory-reserve:zones (Memory Reservation Service)
   State: maintenance since Wed Jan 19 09:52:22 2022
  Reason: Requested by service method: "Invalid service configuration:
  unsupported page size in config/pagesize."
     See: http://support.oracle.com/msg/SMF-8000-AR
     See: man -M /usr/share/man -s 8s memory-reserve
     See: man -M /usr/share/man -s 7 solaris-kz
     See: /var/svc/log/system-memory-reserve:zones.log
  Impact: This service is not running.
  ```

- The following example shows the warning message that is output when the attempt to apply a memory change to the `kz1` kernel zone partially succeeds:

```
# zoneadm -z kz1 apply
Checking: Modifying capped-memory physical=12G
Applying the changes
warning: operation succeeded partially for capped memory (requested: 12G, final:
14G)
```

- The following example shows the error message that is output when the attempt to apply a memory change to the `kz1` kernel zone fails:

  ```
  # zoneadm -z kz1 apply
  Checking: Modifying capped-memory physical=32G
  Applying the changes
  error: Memory reconfiguration not enabled
  error: Modifying capped-memory physical=32G failed
  ```

  In this example, the failure is caused by memory live zone reconfiguration not being enabled for the `kz1` kernel zone.

Strict Method Memory Resize Request Failure

This example uses the strict method to perform a memory resizing operation. First, set the `capped-memory:memlzr` property value to `strict` to enable the strict method. Next, reduce the amount of RAM for the `kz3` kernel zone to 4 Gbytes and apply the changes. This result shows that the commands fail to reduce the RAM size of `kz3` to the requested 4 Gbytes because the strict value is in effect. As a result, the RAM size of the `kz3` kernel zone is unchanged at 16 Gbytes and the `zonecfg apply` command returns `1`.

```
# zonecfg -z kz3 "select capped-memory;set memlzr=strict;end"
# zonecfg -z kz3 info capped-memory
capped-memory:
physical: 16G
pagesize-policy: largest-available
memlzr: strict
# zonecfg -z kz3 "select capped-memory;set physical=4g;end"
# zoneadm -z kz3 apply
Checking: Modifying capped-memory physical=4G Applying the
changes
error: operation could not reach the required memory settings
and the memlzr property was set to 'strict' forcing the
operation to fail. Clear the capped-memory's memlzr property to
accept partial success.
error: Modifying capped-memory physical=4G failed
# echo $?
1
# zlogin kz3
# prtconf | grep Memory
Memory size: 16382 Megabytes
```

5

Migrating an Oracle Solaris Kernel Zone

A zone migration transfers an existing zone from one host system into a zone on another system.

This chapter covers the following topics:

About Kernel Zone Migration

Use the `zoneadm migrate` command to transfer a zone from one system to another. You can migrate kernel zones that are in different states, depending on your requirements. The zone's state determines the type of migration performed.

Oracle Solaris Kernel Zones supports the following types of migration:

- **Cold migration** – The zone is not running on the source host when migrating the zone. The zone's state must be `installed` when you begin the migration, and after migration the state will be the same on the new host.

  This method is useful to migrate zones that use large amounts of memory, or that are providing services that require quick response time, and might not be good use cases for live migration.

- **Warm migration** – You suspend the zone on the source host before migrating the zone. You must configure a `suspend` resource and the zone's state must be `installed` with an auxiliary state of `suspended` when you begin the migration. On the new host after the migration, the zone is also in a `suspended` auxiliary state, so you must resume the zone there.

  This method is useful for zones that are running applications that require a prolonged startup time when the zone is booted and cannot be live migrated. A warm migration requires an outage period for the zone.

- **Live migration** – The zone is actively running on the source host while migrating. The zone's state must be `running` when you begin the migration. The memory state of the migrating zone is copied to the target host during the migration to enable the zone to pick up processing on the new host where it left off.

This method is useful for situations where downtime must be minimized and applications must remain in a `running` state. The migration process can have a performance impact that might negatively affect heavy workloads. In cases where a performance impact is not acceptable, use warm migration or cold migration during a planned outage window.

Migrations must be done by an administrator or a user who has appropriate authorizations. You can enable specific non-root users to perform migrations as described in Rights Required to Perform Kernel Zone Migrations.

Oracle Solaris Kernel Zones also supports migration of all the kernel zones from a host system in a process called **evacuation**. See Evacuating Oracle Solaris Kernel Zones to a Target Host for more information.

Rights Required to Perform Kernel Zone Migrations

A subset of Zones rights profiles enable a non-`root` user to perform kernel zone migrations. If you assign one or more of the following profiles to a user in the global zone, the user can migrate all zones on the system:

- Zone Migration – Enables a zone administrator to perform migration of an installed or running zone.
- Zone Cold Migration – Enables a zone administrator to perform migration of an installed or suspended zone.
- Zone Configuration – Enables a zone administrator to configure the target system for a migrating zone.

For information about Zones rights profiles, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*. To assign rights to migrate zones, see the following examples.

Example 5-1   Authorizing a User to Perform All Migrations of an Individual Zone

```
global1$ pfbash zonecfg -z kzone1
zonecfg:kzone1> add admin
zonecfg:kzone1:admin> set user=jdoe
zonecfg:kzone1:admin> set auths=migrate
zonecfg:kzone1:admin> end
zonecfg:kzone1> commit
Verify the auths and profiles:

global1$ zonecfg -z kzone1 info admin
admin:
user: jdoe
auths: migrate
$ auths jdoe
solaris.admin.wusb.read,solaris.mail.mailq,solaris.network.autoconf.read,solaris.
zone.migrate/kzone1
$ profiles jdoe
jdoe:
Zone Migration
Basic Solaris User
All
```

Example 5-2   Authorizing a User to Migrate All Zones on a Host System

```
global1$ pfbash usermod -P +"Zone Migration" -A +solaris.zone.migrate jdoe
Verify the auths and profiles:
```

```
global1$ auths jdoe
solaris.admin.wusb.read,solaris.mail.mailq,solaris.network.autoconf.read,solaris.zone.m
igrate
global1$ profiles jdoe
jdoe:
Zone Migration
Basic Solaris User
All
```

Example 5-3 Authorizing a User to Configure Zones on a Host System

On the target system, this example assigns the user `jdoe` the required profile and authorization to create the zone configuration. The assigned profile enables the user to create, modify, and delete any zone configuration.

```
global2$ pfbash usermod -P +"Zone Configuration" -A +solaris.zone.config jdoe
Verify the auths and profiles:

global2$ auths jdoe
solaris.admin.wusb.read,solaris.mail.mailq,solaris.network.autoconf.read,solaris.zone.c
onfig
global2$ profiles jdoe
jdoe:
Zone Configuration
Basic Solaris User
All
```

For more information about how to assign and manage rights profiles, refer to Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

Determining a Migration Method to Use

Cold Migration
If the kernel zone is not already running, use cold migration when you need to move the zone to a different host.

Live Migration
If the kernel zone is running and providing services, live migration is typically a good method when moving the zone to a different host. A scheduled outage should not be required, the network connections remain, and the applications can keep running.
However, live migration is not suitable under any one of the following conditions:

- Slow network connection or poor bandwidth between hosts.
- Very large memory configured in the zone.
- Application needs a very low latency response time of less than a few hundred milliseconds. You should test the application to determine this.
- Device is enabled for multi-host disk control operations and the zone configuration includes the `device` property setting `allow-mhd=true`.

Warm Migration
If cold or live migration are not feasible, a warm migration might be a better choice under the following conditions:

- Zone has sufficiently sized suspend LUN available over shared storage.

  The size of the storage for the suspend image should be at least as large as the amount of memory allocated to the kernel zone in its `capped-memory:physical` property.
- Migration occurs during a maintenance window.

  An outage time of up to several minutes can occur.
- Reduced recovery time after migration is desired.

  Warm migration does not use significant network bandwidth. The zone's network connections might not stay up but applications do not require a cold restart.

If neither live or warm migration are feasible, shut down the zone completely and use cold migration.

Kernel Zone Migration Requirements

Live, warm, and cold migration have different requirements.

Requirements for All Kernel Zone Migration Methods

The cold, warm, and live migration methods for kernel zones all have the following requirements:

- The source and target host systems must be the same architecture family. Migrating a zone between SPARC and x86 architectures is not supported.
- Both the source and target host systems must meet the general kernel zone requirements described in Software and Hardware Requirements for Oracle Solaris Kernel Zones.
- The zone storage must be accessible by both the source and target hosts through a shared storage resource defined by an NFS, iSCSI or LU storage URI.

  For more information about NFS storage URIs, see NFS Storage URIs and Kernel Zones.

  For more information about shared storage URIs for other types of storage, see the `suri`(7) man page and Chapter 13, Oracle Solaris Zones on Shared Storage in *Creating and Using Oracle Solaris Zones*
- If you plan to use `ssh` to migrate, SSH public key authentication between the source and target hosts must be configured so that SSH does not require a prompt. In addition to a public key, you must configure `ssh-agent` to remember the public key after you remotely log in at least once to the SSH prompt. See How to Generate a Public/Private Key Pair for Use With Secure Shell in *Managing Secure Shell Access in Oracle Solaris 11.4*.
- The zone configuration must be compatible and consistent on both the source and target hosts. See About Migration and Compatible Configurations.
- If you want to perform warm or live migration of running kernel zones, the source and target hosts must meet additional system requirements described in Additional Requirements for Kernel Zone Warm Migration and Live Migration.

Note:

If you cannot use shared storage, you cannot use the `zoneadm migrate` command to migrate a zone. However, you can create a unified archive and recreate the zone from the archive on a new host system. See *Using Unified Archives for System Recovery and Cloning in Oracle Solaris 11.4*.

Additional Requirements for Kernel Zone Warm Migration and Live Migration

Note:

Also see Additional Requirements for Kernel Zone Live Migration.

In addition to the requirements listed in Kernel Zone Migration Requirements, both source and target systems must meet the following requirements to use warm migration or live migration:

- The operating system on both hosts must be at least Oracle Solaris 11.3.
- If you are migrating a kernel zone between different CPU types in the same platform, you must set the `cpu-arch` property as explained in Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions.
- If you are migrating a kernel zone on a SPARC-based system from Oracle Solaris 11.4 to Oracle Solaris 11.3 and the `host-compatible` property is set, the value must be compatible on the source and target hosts. See Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions.
- If a kernel zone is using SR-IOV (enabled `iov` property), the ability to suspend and resume the kernel zone and migrate using warm or live migration is limited to hosts and zones running a minimum of Oracle Solaris 11.4. See About Migration of Kernel Zones with SR-IOV-Enabled anet Resources for more information.
- Source and target host systems for warm migration and live migration must have running instances of the following services:
  - Remote Administrative Daemon (RAD) services, `svc:/system/rad:local` and `svc:/system/rad:remote`
  - Network Time Protocol (NTP) client

  See Enabling Services for Warm or Live Migration for more information about these services.

Additional Requirements for Kernel Zone Live Migration

In addition to the requirements listed in Kernel Zone Migration Requirements and Additional Requirements for Kernel Zone Warm Migration and Live Migration, both source and target systems must meet the following requirements to use live migration:

- For live migration between SPARC based systems, you must have the following firmware versions installed:

  - SPARC T4 system with at least System Firmware 8.8
  - SPARC T5, SPARC M5, or SPARC M6 system with at least System Firmware 9.5
  - SPARC M8 systems. All firmware versions are supported.
  - SPARC T7 or SPARC M7 series server. All firmware versions are supported.
  - Fujitsu M10/SPARC M10 server. Follow firmware requirements in *Fujitsu M10 Systems Product Notes* that are appropriate for your configuration.
  - Fujitsu SPARC M12 server. All firmware versions are supported.
- Kernel zone live migration service, `svc:/network/kz-migr:stream`. Port 8102 on the target host must be open.
- At least a 10GB Ethernet link is recommended.

  If minimizing downtime is critical, consider limiting other traffic on that link, including other migrations. For more information, see Best Practices for Oracle Solaris Kernel Zones (https://www.oracle.com/technical-resources/articles/it-infrastructure/solaris-kernel-zones-best-practices.html).

Kernel Zone Migration Preparation

Before you migrate kernel zones, review the `zoneadm migrate` command, learn how to make the source and target zones compatible, consider the effects of any differences in CPUs and OS versions, and plan to use the encryption defaults to ensure security during the migration process.

About the `zoneadm migrate` Command

The `zoneadm` command that is used for each migration method is similar. The migration process is determined by the state of the zone that you are migrating. The format of the command is as follows:

```
zoneadm -z kzone migrate [-nqw] -c cipher] [-t {auto|live}] [-w] [[ssh|
rads|radg://user@host:port]
```

where:

-c cipher
Specifies a secure cipher option for 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 Secure Migration.

-n
Performs a non-executing dry run of the migration. For all types of migration, the dry run checks that the shared storage resource is accessible from both systems.

- For live migration, the dry run checks for full compatibility so if it passes the checks, the zone will successfully resume on the remote system.
- For warm migration, even though settings such as CPU and memory must be compatible, the dry run does not check them because the zone is not running during the migration. After the migration, you can adjust the settings as needed before resuming.

- For cold migration and warm migration, the dry run does not check for CPU compatibility or compare the zone configuration on the source system against any existing zone configuration on the target system. For cold migration, there is no need to check for CPU and memory compatibility and so on since the zone is booting from scratch, After the migration, you can adjust the zone configuration settings as needed.

-q
Quiet mode, which specifies that the status is not reported during migration operations.

-t {live | auto}
Specifies the type of migration. The default value of `-t` is `auto` which enables the `zoneadm migrate` command to automatically determine which type of migration is appropriate based on the zone's state. The `-t live` option enables you to restrict migration to running zones only. If you specify `-t live` for a zone that is not running, no migration takes place for that zone.

-w
Uses the zone configuration from the source system. By default, the zone configuration from the source is ignored in favor of the zone configuration on the target. The `-w` option is mutually exclusive with the `-n` option.

`ssh|rads|radg://`*`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 system. The `ssh` scheme uses SSH and the `rads` scheme uses TLS. The `radg` scheme uses the Generic Security Services API (GSS-API). Specify `radg` if the RAD client and target host are configured for Kerberos.
If you specify only 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 in Python to a RAD Instance by Using a URI in *Remote Administration Daemon Client User’s Guide* for more information.

Refer to the `zoneadm`(8) man page for more information about the `migrate` command.

About Migration and Compatible Configurations

A kernel zone's configuration must be completely compatible with the migration target host's environment, as if you were detaching then attaching the zone. A zone that boots on a new host after a warm or live migration is resuming from a saved memory state and is expecting a particular setup. Any incompatibilities cause migration to fail.

If you are migrating a kernel zone to another system that is identical, and all storage references use a storage URI that is accessible by both hosts, the migrated configuration should be compatible without changes.

If the zone storage is local, you cannot use the `zoneadm migrate` command. You can either remove local storage devices from the zone configuration if they are not needed for booting, or convert the storage to shared as described in How to Move a Zone To a Shared Storage Configuration in *Creating and Using Oracle Solaris Zones* and then use `zoneadm migrate`.

Alternatively, you can move the zone using a unified archive. See *Using Unified Archives for System Recovery and Cloning in Oracle Solaris 11.4* for more information.

The following resources and properties must be the same in the zone configuration on the source and target hosts:

- Amount of memory specified for the `capped-memory:physical` value
- Value of the `capped-memory:pagesize-policy` property.
- Number of virtual CPUs specified for `virtual-cpu:ncpus`
- Shared storage URI and `id` for disk devices specified with the `device` resource
- Properties of virtual NICs specified with `net` or `anet` resources

If you configure the zone on the target host before migration, the target host's version of the zone configuration is used to boot the zone. If the configuration is incompatible with the current zone configuration, an error is returned. The encryption keys for the zone must also match. See Encryption Keys and Host Data.

If you do not configure the zone on the target host before migration, the zone configuration is exported from the source host and imported on the target host. The user performing the migration must have the Zone Configuration rights profile and `solaris.zone.configuration` authorization to create zone configurations on the target host. See Rights Required to Perform Kernel Zone Migrations for more information.

If the target host environment is not identical, observe the following guidelines:

- If the CPU of the system is different, you must set the `cpu-arch` to a migration class if you want to do warm or live migration. See Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions. You do not need to set this property for cold migration.
- If the source and target hosts are running different versions of Oracle Solaris on SPARC-based systems, you might need to set `host-compatible` property to specify which Oracle Solaris features can be supported on both hosts. See Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions. You do not need to set this property for cold migration.
- If the source host is running Oracle Solaris 11.4 and the target host is running Oracle Solaris 11.3 you must clear the `pagesize-policy` property. See About Memory Page Size Policy and Physical Memory.

Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions

iKernel zones can be migrated to other host systems that have different CPUs but are the same platform. For example, you can migrate a kernel zone from a SPARC T4 server to a SPARC T7 server, or from an Intel Nehalem-based server to a server based on a Haswell processor. This is called cross-CPU migration. You cannot migrate a zone from a SPARC-based environment to an x86-based environment.

If you want to migrate a kernel zone to a target system whose processor is different from the source system, you must prepare the kernel zone and reboot it *before* you suspend the zone for warm migration or live migrate.

On SPARC-based and Intel-based systems, you can use the `cpu-arch` resource type to specify a migration class that defines a common set of processor features to enable the zone to run well on the source and target system.

The `cpu-arch` resource type is not available for AMD systems.

On SPARC-based systems only, you can additionally set a compatibility level with the `host-compatible` resource type to ensure that Oracle Solaris features that are enabled by specific processors are supported at the same level in the global zones of the source and target host. If `host-compatible` is not set, only Oracle Solaris 11.2 features are visible to the zone. Oracle Solaris 11.2 is the earliest version of Oracle Solaris that can run in a kernel zone, but does not support features such as SSM and DAX.

Note:

The `cpu-arch` and `host-compatible` values in the zone configuration must be supported by the Oracle Solaris release that is running on both source and target hosts in order for live migration to succeed. A zone cannot be live migrated between hosts when a feature is enabled in a different way, even when the feature is supported by both releases. For example, if `host-compatible=adi` is set in the zone configuration on the source host and `host-compatible=level1` is set in the zone configuration on the target host, the live migration fails even if both hosts are running Oracle Solaris 11.4.

See Kernel Zone Migration Class and Host Compatibility Level (solaris-kz Only) in *Oracle Solaris Zones Configuration Resources* for further information about the `cpu-arch` and `host-compatible` resource type properties for migration between different SPARC architectures.

See x86: Cross-CPU Migration Classes (solaris-kz) in *Oracle Solaris Zones Configuration Resources* and SPARC: Cross-CPU Migration Classes (solaris-kz) in *Oracle Solaris Zones Configuration Resources* for more information about setting the `cpu-arch` resource type for migration between different Intel architectures.

If the kernel zone's `cpu-arch` property is not set to a migration class, the kernel zone's CPU architecture is the same as the host system.

Note:

The kernel zone host will always refuse to resume a kernel zone that was previously suspended on an incompatible CPU type. A kernel zone will not boot if the `cpu-arch` class is set to an incompatible value.

Example 5-4 Confirming and Setting the Kernel Zone `cpu-arch` and `host-compatible` Resources on a SPARC Based System

The following example demonstrates how to confirm and set the `cpu-arch` and `host-compatible` resource types on the kernel zone `kzone1`.

```
global$ zonecfg -z kzone1
zonecfg:kzone1> info cpu-arch host-compatible
cpu-arch: generic
host-compatible: native
zonecfg:kzone1> set cpu-arch=migration-class1
zonecfg:kzone1> set host-compatible=adi
zonecfg:kzone1> info cpu-arch host-compatible
cpu-arch: migration-class1
host-compatible: adi
zonecfg:kzone1> exit
```

Example 5-5 Live Migration Fails Due to Incompatible SPARC 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 using a default value that indicates the actual CPU architecture. The `cpu-arch` property value is not consistent across the hosts and must be set to a migration class described in Kernel Zone Migration Class and Host Compatibility Level (solaris-kz Only) 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: Cannot resume guest on target host.
error: Guest's migration class is SPARC-T4, host's is SPARC-T5. Please check cpu-
arch setting in zone config or in host LDom config.
2016-08-18 18:27:53 error: request failed: failed to create VM: Operation not
supported
```

Example 5-6 Confirming and Setting the Kernel Zone Migration Class on an Intel System

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

```
global$ zonecfg -z kzone1
zonecfg:kzone1> info cpu-arch
cpu-arch: generic
zonecfg:kzone1> set cpu-arch=migration-class4
zonecfg:kzone1> info cpu-arch
cpu-arch: migration-class4
zonecfg:kzone1> exit
global$ zoneadm kzone1 reboot
```

Example 5-7 Live Migration Fails Due to Incompatible Intel CPU Architecture

This example demonstrates a live migration attempt between a Haswell-based host `global1` and a Sandy Bridge-based host `global2`. This would occur if you were migrating for example from an Oracle Server X5-2 to a different type of server whose CPU is a Sandy Bridge processor and you did not set `cpu-arch`. The `cpu-arch` property must be set to a migration class described in x86: Cross-CPU Migration Classes (solaris-kz) in *Oracle Solaris Zones Configuration Resources* and SPARC: Cross-CPU Migration Classes (solaris-kz) 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 live migration compatibility.
zoneadm: zone 'kzone1': configuration check failed:
See /var/log/zones/kzone1.messages for debug output
error: cannot resume as current CPU migration class value: sandybridge does not
match the suspended value haswell
2016-08-15 01:05:55 error: request failed: failed to create VM: Invalid argument
```

Secure Migration

By default, 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. *cipher* can be one of the following values:

encryption-cipher
Specifies one of the ciphers that is supported on the source and target hosts.

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

`none`
Disables encryption.

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

Example 5-8   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`.

```
global1$ zoneadm -z kzone1 migrate 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 5-9   Confirming Cipher Compatibility Between Live Migration Source and Destination Hosts

The following example demonstrates an attempt to perform 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 5-10   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: aes-128-cbc
```

```
# echo $?
0
```

Tip:

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

```
global$ pfbash zonecfg -z kzone1 export -f /net/example/path/kzone1.cfg
global$ zonecfg -z kzone1 -f /net/example/path/kzone1.cfg
```

For information about the encryption keys that enable the zone to boot, see Encryption Keys and Host Data.

Using Cold Migration to Migrate a Kernel Zone

In a cold migration, a non-running zone is detached, and attached on another host where you can reboot it.

For more information about cold migration, see About Kernel Zone Migration and Determining a Migration Method to Use.

How to Migrate a Kernel Zone Using Cold Migration

Ensure that the source and target hosts meet requirements described in Kernel Zone Migration Requirements.

1. Become an administrator who is assigned rights to migrate kernel zones.

   For more information, see Rights Required to Perform Kernel Zone Migrations.

2. If the zone to migrate is running, shut it down.

   ```
   source-host$ zoneadm -z kzone shutdown
   ```

3. Verify that the state is installed.

   Output is similar to the following:

   ```
   source-host$ zoneadm -z kzone list -v
   ID NAME             STATUS      PATH      BRAND           IP
   -  kzone           installed    -        solaris-kz       excl
   ```

4. If you plan to use the `ssh://` URI to connect to the target host, test SSH promptless authentication.

   Execute a command such as `date` through `ssh` on the target host.

   ```
   source-host$ ssh target-host date
   Tue Oct  4 17:07:55 MDT 2016
   ```

   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 11.4*.

5. Perform a dry run of the migration to verify that conditions are set appropriately.

   Output is similar to the following:

   ```
   source-host$ zoneadm -z kzone migrate -n ssh://user@target-host
   zoneadm: zone 'kzone': Importing zone configuration.
   zoneadm: zone 'kzone': Attaching zone.
   zoneadm: zone 'kzone': Dry-run migration successful.
   zoneadm: zone 'kzone': Cleaning up.
   ```

6. Perform the migration.

   Output is similar to the following:

   ```
   source-host$ zoneadm -z kzone migrate ssh://user@target-host
   zoneadm: zone 'kzone': Importing zone configuration.
   zoneadm: zone 'kzone': Attaching zone.
   zoneadm: zone 'kzone': Migration successful.
   ```

7. Boot the zone on the target host.

   ```
   target-host$ zoneadm -z kzone boot
   ```

Example 5-11 Dry Run Fails for Cold Migration Due to Local Storage

The following example shows verification that a kernel zone is not running, and then a cold migration dry run that fails because of local storage being used in the kernel zone `z3kz`.

```
root@global3 $ zoneadm list -cv
ID NAME             STATUS      PATH                           BRAND      IP
 0 global           running     /                              solaris    shared
 - z3kz             installed   -                              solaris-kz excl
root@global3 $ zoneadm -z z3kz migrate -n ssh://global5
zoneadm: zone 'z3kz': configuration check failed: The storage property
dev:/dev/zvol/dsk/rpool/VARSHARE/zones/z3kz/disk0  is not a shared storage URI.
```

Example 5-12 Successful Cold Migration After Configuration Change

The following example shows removal of a non-booting local storage device from the zone configuration for zone `z3kz`, and then a successful migration. The zone configuration that is used comes from the migrating zone and is imported on the target host.

```
root@global3 $ zonecfg -z z3kz 'remove device id=1;commit;exit'
root@global3 $ zoneadm -z z3kz migrate ssh://global5
zoneadm: zone 'z3kz': Importing zone configuration.
zoneadm: zone 'z3kz': Attaching zone.
zoneadm: zone 'z3kz': Migration successful.
```

If you are migrating to an earlier version of Oracle Solaris, see Clearing the pagesize-policy Property Before Migrating a Kernel Zone to an Earlier Oracle Solaris Release.

Enabling Services for Warm or Live Migration

As stated in Kernel Zone Migration Requirements, several services must be running on the source and target hosts when you perform warm or live migration. This section describes how to check the required services and enable them if necessary.

For additional information about managing Oracle Solaris services, see Chapter 3, Administering Services in *Managing System Services in Oracle Solaris 11.4*.

How to Check and Enable Services Needed to Migrate Kernel Zones

1. Become a zone administrator.

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. If you plan to use the ssh:// URI to connect to the target host, test SSH promptless authentication.

   Execute a command such as `date` through `ssh` on the target host.

   ```
   source-host$ ssh target-host date
   Tue Oct  4 17:07:55 MDT 2016
   ```

   If you are prompted for a password, you have not completely configured your SSH login.

   See How to Generate a Public/Private Key Pair for Use With Secure Shell in *Managing Secure Shell Access in Oracle Solaris 11.4*.

3. Check that the NTP client service is running.

   ```
   source-host$ svcs ntp
   STATE          STIME    FMRI
   online         Jun_27   svc:/network/ntp:default

   source-host$ ssh target-host svcs ntp
   STATE          STIME    FMRI
   online         Aug_09   svc:/network/ntp:default
   ```

   If the NTP service is not running, start the NTP service. See How to Set Up NTP on a Oracle Solaris System in *Managing Clock Synchronization in Oracle Solaris 11.4* for additional information.

4. Check that RAD services are running on source and target hosts.

   - If you plan to use the `ssh://` URI, the `rad:local` service must be running.
   - If you plan to use `rads://` or `radg://`, the `rad:remote` service must be running.

   a. Display the status of RAD services on the source and target hosts.

      The following example output shows the `rad:remote` service disabled on both the source and target hosts:

      ```
      Check RAD services on the source host:
      source-host$ svcs rad
      STATE          STIME    FMRI
      disabled       10:09:15 svc:/system/rad:remote
      online         10:09:18 svc:/system/rad:local
      online         10:09:18 svc:/system/rad:local-http

      Check RAD services on the target host:
      source-host$ ssh target-host svcs rad
      STATE          STIME    FMRI
      disabled       Jun_23   svc:/system/rad:remote
      online         Jun_23   svc:/system/rad:local
      online         Jun_23   svc:/system/rad:local-http
      ```

   b. If a RAD service is disabled, enable the service.

The following example commands enable the `rad:remote` service on both the source and target hosts:

```
Enable RAD services on the source host:
source-host$ svcadm enable rad:remote
Enable RAD services on the target host:
source-host$ ssh target-host svcadm enable rad:remote
```

c. Repeat Step 4.a to verify that all RAD services are enabled.

5. If you plan to perform live migration, start the kernel zone migration service on the source and target hosts.

```
source$ svcadm enable -rs svc:/network/kz-migr:stream
source$ ssh target svcadm enable -rs svc:/network/kz-migr:stream
```

Using Warm Migration to Migrate a Kernel Zone

You can migrate a kernel zone to another host by using the `zoneadm suspend` command followed by the `zoneadm migrate` command. 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 the zone configurations to be compatible on both the source and target hosts. If you create a zone configuration on the target host before migration, that configuration is used. Otherwise the configuration of the migrating zone is used.

Warm migrations require that the zone has a `suspend` resource configured to have shared storage accessible by both the source and target hosts. See the `solaris-kz`(7) man page and Chapter 13, Oracle Solaris Zones on Shared Storage in *Creating and Using Oracle Solaris Zones*.

The migration automatically verifies that the zone's shared storage is accessible from the target system, detaches the kernel zone on the source system, and attaches the zone on the destination system.

Note:

If you are migrating systems that are not using identical processors, before suspending, set the `cpu-arch` property as explained in Preparation for Migrating Kernel Zones to Systems With Different CPUs or OS Versions.

You can resume the zone manually on the target system after the migration is complete. See About Resuming a Kernel Zone After a Warm Migration for information about resuming.

How to Migrate a Kernel Zone by Using Warm Migration

- Ensure that the source and target hosts meet requirements described in Kernel Zone Migration Requirements.
- Ensure that the required services are available as described in Enabling Services for Warm or Live Migration.

- If the source and target hosts are not identical, see About Migration and Compatible Configurations.

1. Become an administrator who is assigned rights to migrate kernel zones.

   For more information, see Rights Required to Perform Kernel Zone Migrations.

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

   Output is similar to the following:

   ```
   source-host$ zonecfg -z kzone info suspend
   suspend:
           storage: iscsi://system/luname.naa.501337600144f0dbf8af1900
   ```

   The target host must have access to this location using the same URI. If the `suspend` resource is not configured, see Configuring the suspend Resource Type.

3. On the global zone of the source host, suspend the kernel zone to be migrated.

   ```
   source-host$ zoneadm -z kzone suspend
   ```

   > **Note:**
   >
   > The suspend process can be time-consuming, as it writes the zone state including memory to disk.

4. Perform a dry run of the migration and verify that shared storage is accessible.

   Output is similar to the following:

   ```
   source-host$ zoneadm -z kzone migrate -n ssh://user@target-host
   zoneadm: zone 'kzone': Importing zone configuration.
   zoneadm: zone 'kzone': Attaching zone.
   zoneadm: zone 'kzone': Dry-run migration successful.
   zoneadm: zone 'kzone': Cleaning up.
   ```

   If the dry run reveals problems with the shared storage accessibility, correct them before proceeding. See Chapter 13, Oracle Solaris Zones on Shared Storage in *Creating and Using Oracle Solaris Zones*.

5. Migrate the zone to the target host.

   This step configures a zone on the target host system using the same zone configuration on the source host, and attaches the zone on the target host.

   ```
   source-host$ zoneadm -z kzone migrate rad-uri:user@target-host
   ```

   - Use `ssh` to migrate:

     ```
     source-host$ zoneadm -z kzone migrate ssh://root@target-host
     zoneadm: zone 'kzone': Importing zone configuration.
     zoneadm: zone 'kzone': Attaching zone.
     zoneadm: zone 'kzone': Migration successful.
     ```

   - Use `rads` to migrate:

     ```
     source-host$ zoneadm -z kzone migrate rads://root@target-host:12302
     ```

6. Boot the kernel zone on the new host to resume the migrated zone.

   ```
   target-host$ zoneadm -z kzone boot
   ```

Example 5-13 Failed Configuration Check on Warm Migration

This example shows a migration attempt with the suspend resource on a local path.

```
global3$ zoneadm -z z1kz migrate ssh://global5
zoneadm: zone 'z1kz': configuration check failed: suspend path resource must be an NFS
path
```

Example 5-14 Suspending the Kernel Zone and Warm Migrating

This example shows the commands for displaying the `suspend` resource, suspending the zone, listing the zones to include the kernel zone auxiliary state which is `suspended` for `z2kz`, and then the successful migration of the suspended kernel zone. Issuing the `uptime` command on the migrated zone on the target host shows how long it has been running, which includes time since it was booted on the source host.

```
global3$ zonecfg -z z2kz info suspend
suspend:
         storage: iscsi://system/luname.naa.501337600144f0dbf8af1900
global3$ zoneadm -z z2kz suspend

global3$ zoneadm list -cp
0:global:running:/::solaris:shared:-:none:
-:z2kz:installed:/system/volatile/zones/z2kz/zonepath:890d94b7-23c7-48c8-922e-
ede10c3d1ac6:solaris-kz:excl:-:solaris-kz:suspended

global3$ zoneadm -z z2kz migrate ssh://global5
zoneadm: zone 'z2kz': Importing zone configuration.
zoneadm: zone 'z2kz': Attaching zone.
zoneadm: zone 'z2kz': Migration successful.

global3$ ssh global5 zlogin z2kz uptime
12:02pm  up 2 day(s),  2:55,  0 users,  load average: 0.04, 0.04, 0.03
```

About Resuming a Kernel Zone After a Warm Migration

The migrated zone might fail to boot if the zone configuration on the source system was modified after the zone was suspended or halted. Also, live zone configuration changes might have been made to the running zone before it was suspended that are not reflected in the permanent zone configuration. For this reason, no compatibility checking is done between source and target zone configurations or systems. Only the storage is checked.

You should be able to resume a migrated kernel zone on a target system if you ensured before migration that the configuration was compatible. If a suspended zone fails to resume on the target system due to compatibility or zone configuration differences between the source and target systems, modify the zone configuration on the target system and then try again to resume. You can also boot the zone in a non-resume mode by specifying the -R option with the `boot` command, but this results in the saved information in the zone's suspended image not being used, and it is no longer a warm migration.

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 zone, a live migration results in a brief outage time that is not noticeable to most applications or to most end users. Network connections are maintained.

If the kernel zone that you want to migrate uses a memory reservation pool (MRP), ensure that MRP is enabled on the target host. See Configuring a Memory Reservation Pool for Kernel Zones.

You can use live migration for any applications that require a minimum of downtime and where applications must continue providing service without interruption.

How to Migrate a Kernel Zone By Using Live Migration

- Ensure that both the kernel zone source and target hosts meet hardware, software, and storage requirements for live migration as detailed in Kernel Zone Migration Requirements and Additional Requirements for Kernel Zone Warm Migration and Live Migration.
- Ensure that the required services are available as described in Enabling Services for Warm or Live Migration.
- If the source and target hosts are not identical, see About Migration and Compatible Configurations.

1. Become an administrator who is assigned rights to migrate kernel zones.

   For more information, see Rights Required to Perform Kernel Zone Migrations.

2. On the source host, confirm that the zone to migrate is in the running state.

   ```
   source-host$ zoneadm list -cv
     ID NAME             STATUS     PATH      BRAND            IP
      0 global           running     /        solaris          shared
      1 kzone            running     -        solaris-kz       excl
   ```

3. On the source host, initiate a dry run.

   This operation tests the kernel zone configuration before performing the live migration. Output is similar to the following:

   ```
   source-host$ zoneadm -z kzone migrate -n rad-uri:user@target-host
   zoneadm: zone 'kzone': Importing zone configuration.
   zoneadm: zone 'kzone': Attaching zone.
   zoneadm: zone 'kzone': Booting zone in 'migrating-in' mode.
   zoneadm: zone 'kzone': Checking migration compatibility.
   zoneadm: zone 'kzone': Cleaning up.
   zoneadm: zone 'kzone': Dry-run migration successful.
   ```

4. Migrate the kernel zone.

   ```
   source-host$ zoneadm -z kzone migrate rad-uri:user@target-host
   zoneadm: zone 'kzone': Importing zone configuration.
   zoneadm: zone 'kzone': Attaching zone.
   zoneadm: zone 'kzone': Booting zone in 'migrating-in' mode.
   …
   zoneadm: zone 'kzone1': Halting and detaching zone on source host.
   zoneadm: zone 'kzone': Migration successful.
   ```

5. Confirm that the zone has migrated on the target host.

   ```
   target-host:$ zoneadm list -cv
     ID NAME             STATUS     PATH        BRAND         IP
      0 global           running    /           solaris       shared
   1754 kzone            running    -           solaris-kz    excl
   …
   ```

Example 5-15 Using Live Migration to Migrate a Kernel Zone to a New Host

The following example demonstrates a live migration of the kernel zone `kzone1` from the source host `global` to the target host `global2`. The configuration was created in advance on the target host.

```
global$ zoneadm -z kzone1 migrate ssh://global2
zoneadm: zone 'kzone1': Using 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': Waiting for migration to complete.
zoneadm: zone 'kzone1': Migration successful.
zoneadm: zone 'kzone1': Halting and detaching zone.
```

Example 5-16 Live Migration Dry Run Failure

This example demonstrates a failed dry-run migration 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(8) on
the zone 'kzone1': destination host.
zoneadm: zone kzone1: call to zoneadmd(8) failed: zoneadmd(8) returned an error 9
(zone state change failed)
```

Example 5-17 Live Migration Between Hosts With Different `anet` Configurations

The following example demonstrates live migration between hosts with different `anet` configurations. See *Oracle Solaris Zones Configuration Resources* for additional information regarding `anet` resources.

The zone configuration is created on the target host and the `anet` resource is modified before the migration. A dry run is performed to test.

```
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
```

6

Evacuating Oracle Solaris Kernel Zones to a Target Host

Kernel zone evacuation enables you to transfer all kernel zones in either the `running` or `installed` state from one system to another system. A primary use for evacuation is zero-downtime system maintenance. This chapter covers kernel zone evacuation requirements and tasks.

About Kernel Zone Evacuation

Kernel zone **evacuation** is the process of live migrating all the running kernel zones off a system at once, and optionally returning them to the system later. By live migrating all kernel zones from a host system onto other systems temporarily, you can perform maintenance on the host system without having to halt applications that are running in those kernel zones. You can optionally evacuate all zones, including kernel zones that are not running `solaris` zones in the `installed` state.

Evacuation uses the Remote Administration Daemon (RAD) to coordinate and execute the migration of zones to their destinations, so RAD services must be running on source hosts and target hosts. For more information about RAD, see the `rad`(8) man page.

Kernel Zone Evacuation Steps

The overall process for kernel zone evacuation is:

1. Ensure the requirements are met for the source and target hosts.

   Go to Requirements for Kernel Zone Evacuation.

2. Set a destination host for each of the migrating kernel zones by setting an SMF service property.

   Go to Setting the Target Host for Kernel Zone Evacuation.

3. Place the source host in maintenance mode to prevent non-running zones from attaching, booting, or migrating in.

   Go to Setting Maintenance Mode to Prepare for Kernel Zone Evacuation.

4. Run the `evacuate` command to migrate the running kernel zones to their preset destination.

   Go to Evacuating Kernel Zones.

5. Perform the system maintenance on the source host and reboot.

6. End the maintenance mode on the source host system.

   Go to Ending Maintenance Mode After Kernel Zone Evacuation.

7. Return the evacuated zones to the source host, if desired. Evacuation can also be used to permanently migrate zones to a new host.

   Go to Returning Evacuated Kernel Zones to the Original System.

`sysadm` Utility and Oracle Solaris Kernel Zones

The `sysadm` utility enables you to perform all evacuation tasks.

- Use the `sysadm maintain` command to prepare for migration by placing the system in a maintenance mode that prevents zones from being attached, booted, or migrating in.
- Use the `sysadm evacuate` command to perform the actual migration of zones in an evacuation operation.

See the `sysadm`(8) man page for complete information.

`sysadm maintain` Command

The `sysadm maintain` command accepts the following options:

-e
End maintenance mode.

-l
List the current status of maintenance mode.

-m "*message-text*"
Message you can specify to indicate reason for maintenance, for example.

-s
Start maintenance mode to prevent new zones from running on the source host.

Maintenance state persists across system reboots and applies to all boot environments, so you must explicitly end the maintenance mode with the `sysadm maintain -e` command.

`sysadm evacuate` Command

The `sysadm evacuate` command accepts the following options:

-a
All zones, both non-running and running, are evacuated.

-n
Non-executing dry run of the evacuation to show how the evacuation would be performed.

-q
Quiet output, showing only errors.

-r
Return the evacuated zones to the source host.

-v
Verbose output, showing progress of the evacuation.

-w
Overwrite zone configurations of all zones on the destination host with the respective configurations from the source host. This option is mutually exclusive with the -n option.

Requirements for Kernel Zone Evacuation

Observe the following requirements to perform kernel zone evacuations:

- **Migration requirements** – The source and target hosts and their configured storage must meet the requirements described in Kernel Zone Migration Requirements.
- **Oracle Solaris version** – For full support of all features, including evacuation and return of running kernel zones and non-running installed kernel zones and non-global zones, both source and target hosts must be running Oracle Solaris 11.4. Zones that are in the `installed` state are evacuated using cold migration.

  Evacuation and return of running kernel zones from systems running Oracle Solaris 11.4 to systems running Oracle Solaris 11.3 is also supported. However, because cold migration is not supported in Oracle Solaris 11.3, you cannot use the -a option to evacuate zones that are not running.
- **Live migration requirements** – The evacuation process uses live migration, so the source and target hosts must also meet the requirements for live migration. See the Additional Requirements for Kernel Zone Warm Migration and Live Migration for more information.
- **Enabled services** – You must enable services as described in Enabling Services for Warm or Live Migration.
- **User rights and authorizations** – The user performing the evacuation must have rights profiles and authorizations as described in Rights Required to Perform Kernel Zone Migrations. In addition, the user must have the Maintenance and Repair rights profile to run the `sysadm maintain` command.
- **Non-interactive authentication** – You must configure non-interactive authentication through a RAD transport between the source host and each destination in both directions. That is, the user performing the evacuation must be able to connect from source host to target host and from target host to source host using a RAD URI without being prompted for a password or to confirm key fingerprints. You may need to ensure that you can connect non-interactively using both the simple host name and the fully qualified host name, which includes the domain name.

  SSH public key authentication between the source and target hosts is one way to perform non-interactive authentication. See How to Generate a Public/Private Key Pair for Use With Secure Shell in *Managing Secure Shell Access in Oracle Solaris 11.4* for more information.

Setting the Target Host for Kernel Zone Evacuation

After the non-interactive authentication between the hosts is set up, set the `evacuation/target` SMF property of the zones delegated restarter service to specify the URI to connect to the RAD service on the target host.

You must select a target host that is suitable to receive and run the evacuated zones. If any migrating zone will need any changes in its configuration to run on the target host, you should create the zone configuration on the target host before evacuation. See About Migration and Compatible Configurations.

You can set the `evacuation/target` property to apply to all zones on the source host, or set the property to different target hosts for individual zones. You can also override the setting on individual zone service instances after you set the property for all zones.

You must refresh the zone delegated restarter service `svc:/system/zones/zone` after setting the `evacuation/target` property for all zones. If you set a target for a particular zone, you must refresh its zone delegated restarter service `svc:/system/zones/zone:zonename` after setting the `evacuation/target` property.

To evacuate all zones to the same host *targethost*:

```
global$ pfbash svccfg -s system/zones/zone
svc:/system/zones/zone> setprop evacuation/target=ssh://targethost

svc:/system/zones/zone> exit
global$ svcadm refresh svc:/system/zones
```

To evacuate a particular zone *zonename* to a target host *targethost* add the zone name to the service name:

```
global$ pfbash svccfg -s system/zones/zone:zonename

svc:/system/zones/zone:zonename> setprop evacuation/target=ssh://targethost

svc:/system/zones/zone:zonename> exit
global$ svcadm refresh svc:/system/zones/zone:zonename
```

Example 6-1 Setting the Evacuation Target for All Zones and Verifying the Property Value

```
global$ pfbash svccfg -s system/zones/zone
svc:/system/zones/zone> setprop evacuation/target=ssh://global2
svc:/system/zones/zone> exit global$ svcprop -p "evacuation/target" svc:/system/
zones/zone
ssh://global2
global$ svcadm refresh svc:/system/zones/zone
```

Example 6-2 Setting the Evacuation Target for an Individual Kernel Zone

```
global$ pfbash svccfg -s system/zones/zone:kz1
svc:/system/zones/zone:kz1> setprop evacuation/target=ssh://global2
svc:/system/zones/zone:kz1> exit
global$ svcprop -p "evacuation/target" svc:/system/zones/zone:kz1
ssh://global2global$ svcadm refresh svc:/system/zones/zone:kz1
```

Setting Maintenance Mode to Prepare for Kernel Zone Evacuation

Use maintenance mode to prepare for evacuation of the zones. Starting maintenance mode logs an audit record, and prevents the subsequent attach, boot, or incoming migration of any zones into the system. You can start maintenance mode before you perform administrative tasks on a zones host to remove it from service before you migrate the zones.

The Maintenance and Repair rights profile is required to change maintenance state.

To place the source host in maintenance mode and specify an optional accompanying message, using the following command:

```
sysadm maintain -s -m "message-text"
```

For example, to place a system in maintenance mode with a message and then list the maintain status:

```
global$ pfbash sysadm maintain -s -m "Updating system to new release"
global$ sysadm maintain -l
TYPE   USER     DATE               MESSAGE
admin  root     2016-07-22 17:57   Updating system to new release
```

The status shows the type of maintenance, the user who ran the `maintain` command, the time run, and the message. The only type of maintenance currently supported is `admin`.

Evacuating Kernel Zones

After you have set up non-interactive authentication, set the evacuation targets, and initiated maintenance mode, you can proceed with zone evacuation.

The default `evacuate` command attempts to live migrate each running zone to the evacuation target configured in the zone's SMF service instance's `evacuation/target` property. If a running zone is a brand other than `solaris-kz`, it cannot be live migrated, and evacuation is skipped for that zone.

You can specify the -a option to evacuate all zones including non-running zones which will be evacuated using cold migration.

You can specify the -n option to perform a non-executed or dry-run evacuation. Migration is planned, and a dry-run migration to the target host is performed for each zone and the outcome or errors are reported.

Evacuation can be run several times in case some zones fail to evacuate on the first run. The return status is successful only if the evacuation as a whole is complete and no zones are running on the source host after executing the `sysadm evacuate` command.

How to Evacuate Running Kernel Zones

Be sure the evacuation targets are set and the system is in maintenance mode as described in the preceding sections.

1. Become an administrator who is assigned rights to migrate kernel zones.

   For more information, see Rights Required to Perform Kernel Zone Migrations.

2. Verify that the zones that you want to evacuate are running.

   Output is similar to the following:

   ```
   source-host$ pfbash zoneadm list -cv
   ID NAME             STATUS     PATH                           BRAND      IP
    0 global           running    /                              solaris    shared
    1 kzone            running    -                              solaris-kz excl
   ```

3. Perform a dry run of the evacuation to verify that conditions are set appropriately.

   Output is similar to the following:

```
source-host$ sysadm evacuate -n
sysadm: preparing zones for evacuation ... 1/1
sysadm: dry-run succeeded
```

4. Evacuate the zones.

   Output is similar to the following:

```
source-host$ sysadm evacuate -v
sysadm: preparing 3 zone(s) for evacuation ...
sysadm: initializing migration of kzone to new-host ...
…
sysadm: evacuation completed successfully.
```

Example 6-3 Successful Zone Evacuation

```
root@ldom1-04:~# sysadm evacuate -v
sysadm: preparing 1 zone(s) for evacuation ...
sysadm: initializing migration of kzone1 to ldom1-08 ...
sysadm: evacuating 1 zone(s) ...
sysadm: migrating kzone1 to ldom1-08 ...
sysadm: evacuation completed successfully.
sysadm: kzone1: evacuated to ssh://ldom1-08
```

Example 6-4 Native Zone Skipped When Evacuating Zones

This example shows two kernel zones successfully live migrated in an evacuation and one `solaris` brand zone skipped, resulting in a "failed" evacuation as a whole. The kernel zones were successfully evacuated however.

```
root@global :~# sysadm evacuate -v
sysadm : preparing 3 zone(s) for evacuation ...
sysadm : initializing migration of kzone1 to global2 ...
sysadm : initializing migration of kzone2 to global2 ...
sysadm : evacuating 2 zone(s) ...
sysadm : migrating kzone1 to global2 ...
sysadm : migrating kzone2 to global2 ...
sysadm : evacuation failed .
sysadm : kzone1: evacuated to ssh://global2
sysadm : kzone2: evacuated to ssh://global2
sysadm : my-ngz : evacuation skipped: cannot evacuate solaris-brand zones
```

Checking Zone Evacuation Status

Use the `sysadm evacuate -l` command to check the status of an evacuation.

```
root@global :~# sysadm evacuate -l
ZONENAME         STATE     DEST                                     ERROR
kzone1           EVACUATED ssh://global2                            -
kzone2           EVACUATED ssh://global2                            -
my-ngz           SKIPPED
```

Zones that are successfully evacuated have a state of `EVACUATED`.

A running non-global zone cannot be live migrated, and evacuation is skipped for that zone, so the state for that zone is `SKIPPED`.

If an individual zone fails to evacuate, the state for that zone is `FAILED`, and the `ERROR` value provides more information.

Ending Maintenance Mode After Kernel Zone Evacuation

When the administrative tasks are complete and you have rebooted if necessary, you should clear the maintenance mode. This applies whether or not you intend to return the evacuated zones. You cannot boot zones while maintenance mode is in effect.

To end maintenance mode:

```
root@global :~# sysadm maintain -e
```

Returning Evacuated Kernel Zones to the Original System

To return the evacuated zones to the original system, run the `sysadm evacuate` command on the source host, *not* the target host. You specify the -r option to return.

Each evacuated zone is migrated from its destination, if it is still running there, back to the source host. If you want to also return zones that are not running, use the -a option to specify all zones. Non-running zones are cold migrated.

```
source-host# sysadm evacuate -rv
```

Example 6-5 Returning Evacuated Zones to Original Host System

```
root@global:~# sysadm evacuate -rv
sysadm:   preparing 2 zone(s) for return...
sysadm:   initializing return of kzone1
sysadm:   initializing return of kzone2
sysadm:   returning 2 zone(s) ...
sysadm:   migrating kzone1
sysadm:   migrating kzone2
sysadm:   return completed successfully.
sysadm:   kzone1: returned
sysadm:   kzone2: returned
root@global:~# sysadm evacuate -l
sysadm:   no active evacuation
root@global:~# zoneadm list -cv
ID NAME             STATUS      PATH                   BRAND      IP
 0 global           running     /                      solaris    shared
 3 kzone2           running     -                      solaris-kz excl
 4 kzone1           running     -                      solaris-kz excl
```

Kernel Zone Evacuation Example

This section annotates the evacuation of two kernel zones.

Example 6-6 Complete Process for Evacuating Kernel Zones

```
List the zones with verbose output to determine status

root@global:~# zoneadm list -v
ID NAME             STATUS      PATH                         BRAND       IP
 0 global           running     /                            solaris     shared
17 kzone2           running     -                            solaris-kz  excl
18 kzone1           running     -                            solaris-kz  excl
19 my-ngz           running     /system/zones/my-ngz         solaris     excl
```

```
Set the evacuation/target SMF property for zones restarter service
to specify migrated zones' destination and refresh the zones restarter service

root@global:~# svccfg -s system/zones/zone
svc:/system/zones/zone> setprop evacuation/target=ssh://global2
svc:/system/zones/zone> exit
root@global:~# svcprop -p "evacuation/target" svc:/system/zones/zone
ssh://global2
root@global:~# svcadm refresh svc:/system/zones

Put the system in maintenance mode with a message

root@global:~# sysadm maintain -s -m "Updating to new release"
root@global:~# sysadm maintain -l
TYPE   USER     DATE               MESSAGE
admin  root     2016-03-16 17:57   Updating to new release

Evacuate the zones with verbose output

root@global :~# sysadm evacuate -v
sysadm : preparing 3 zone(s) for evacuation ...
sysadm : initializing migration of kzone1 to global2 ...
sysadm : initializing migration of kzone2 to global2 ...
sysadm : evacuating 2 zone(s) ...
sysadm : migrating kzone1 to global2 ...
sysadm : migrating kzone2 to global2 ...
sysadm : evacuation failed .
sysadm : kzone1: evacuated to ssh://global2
sysadm : kzone2: evacuated to ssh://global2
sysadm : my-ngz: evacuation skipped: cannot evacuate solaris-brand zones

List evacuated zones

root@global :~# sysadm evacuate -l
ZONENAME STATE     DEST            ERROR
kzone1   EVACUATED  ssh://global2    -
kzone2   EVACUATED  ssh://global2    -
my-ngz   SKIPPED    -              cannot evacuate solaris-brand zones

root@global :~# zoneadm -z  my-ngz shutdown

Perform maintenance such as updating the system

root@global :~# pkg update ...
root@global :~# reboot ...

End maintenance mode

root@global :~# sysadm maintain -e

Return evacuated zones

root@global:~# sysadm evacuate -rv
sysadm:   preparing 2 zone(s) for return...
sysadm:   initializing return of kzone1
sysadm:   initializing return of kzone2
sysadm:   returning 2 zone(s) ...
sysadm:   migrating kzone2
sysadm:   migrating kzone1
sysadm:   return completed successfully.
sysadm:   kzone1: returned
```

```
sysadm:   kzone2: returned

Check evacuation status

root@global:~# sysadm evacuate -l
sysadm:   no active evacuation

Check zone status

root@global:~# zoneadm list -cv
  ID NAME             STATUS      PATH                 BRAND      IP
   0 global           running     /                    solaris    shared
   3 kzone2           running     -                    solaris-kz excl
   4 kzone1           running     -                    solaris-kz excl
```

7

Administering Oracle Solaris Kernel Zones

This chapter covers the following topics:

- Working in the Kernel Zone Environment
- Working With Immutable Kernel Zones
- Managing Removable Devices on the Kernel Zone
- Kernel Zone Auxiliary States Viewed From the Global Zone
- Managing Non-Global Zones in Kernel Zones
- Kernel Zone Host Data and Host ID
- Invoking the Kernel Zone Boot Loader
- NFS Storage URIs and Kernel Zones
- Core Files in Kernel Zones

For information about administrative topics for `solaris` and `solaris10` branded zones, see Chapter 7, About Non-Global Zone Administration in *Creating and Using Oracle Solaris Zones*. Kernel zones do not support `solaris10` branded zones.

Working in the Kernel Zone Environment

Working in a kernel zone environment is very similar to working in a global zone. The following sections describe the major differences between the kernel zone administrative environment and working with a global zone.

Displaying Kernel Zone Process Information

Kernel zone processes are not directly visible to the global zone or the kernel zone host system. You must use the `zlogin` command followed by a process management command to view any process information about a kernel zone, as follows:

```
global$ pfbash zlogin -z kernel-zone process-management-command
```

For example, the following command displays process information about the `syslogd` daemon on the kernel zone `kzone1` from the kernel zone host system `global`:

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

Duplicate Process IDs in a Kernel Zone and Its Global Zone

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`, type the `zlogin` command followed by the `kill` command.

```
global$ pfbash 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 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 Immutable Kernel Zones

Immutable Zones enforce mandatory write access control (MWAC), which provides read-only, or immutable, file system security. Oracle Solaris 11.4 supports immutable zones on global zones, kernel zones, and non-global zones. For detailed information, see Chapter 10, Configuring and Administering Immutable Zones 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. See How to Add a Virtual CD-ROM Device to a Kernel Zone for an example of the process.

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

Perform this task to add a virtual CD-ROM device to a kernel zone.

1. Become a zone administrator.

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

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

   ```
   global$ pfbash lofiadm -r
   ```

   The following example shows sample output.

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

3. Add the lofi device to the kernel zone.

   Output is similar to the following:

```
global$ zonecfg -z kzone
zonecfg:kzone> add device
zonecfg:kzone:device> set storage=dev:path-to-device
zonecfg:kzone:device> end
zonecfg:kzone> exit
```

4. Reboot the kernel zone to apply the configuration changes.

   ```
   global$ zoneadm -z kzone reboot
   ```

5. Log in to the kernel zone.

   ```
   global$ zlogin kzone
   ```

6. On the kernel zone, update the device file system (devfs) and restart the hardware abstraction layer (hal).

   This step enables the `hal` service to see the virtual CD-ROM device.

   ```
   kzone$ pfbash devfsadm -i zvblk
   kzone$ svcadm restart hal
   ```

7. On the kernel zone, list the removable devices.

   Output is similar to the following:

   ```
   kzone$ 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>
   …
   ```

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 CD-ROM image `/root/sol-11_3-repo.full.iso` to 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. In the kernel zone, mount the CD-ROM device.

   ```
   kzone$ mount -F hsfs device-location /mnt
   ```

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

    ```
    kzone$ umount /mnt
    ```

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

    ```
    kzone$ eject cdrom
    ```

12. In the global zone, verify that the ISO image is no longer associated with the lofi device.

    Output is similar to the following:

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

Kernel Zone Auxiliary States Viewed From the Global Zone

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 or migrate zones.

The kernel zone auxiliary states are as follows:

`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 2, Setting Up a Non-Global Zone in *Creating and Using Oracle Solaris Zones*.

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

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

`no-config`
The zone is known to the system, but its configuration is missing. State of the zone is always `incomplete`.

`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.

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

To view the global zone current state and the kernel zone auxiliary states, use the `zoneadm list -s` command. Output is similar to the following:

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

For additional information about kernel zone auxiliary states, see the `solaris-kz`(7) man page.

For information about non-global zone states, see Chapter 1, Oracle Solaris Zones Introduction in *Introduction to Oracle Solaris Zones*.

For information about the kernel debugger see the `kmdb`(1) man page.

Managing Non-Global Zones in Kernel Zones

This section discusses the zone requirements, MAC address configuration, and other zone management issues for non-global zones nested in a kernel zone.

Requirements for Native Zones in Kernel Zones

A kernel zone is the only type of zone that can serve as the global zone to non-global zones, specifically `solaris` zones. You can create, install, and boot `solaris` branded non-global zones inside a kernel zone. You cannot create a kernel zone inside another kernel zone. Zones that run in kernel zones are sometimes called nested zones or hierarchical zones.

A `solaris` zone that runs in a kernel zone must meet the following requirements:

Operating System
The kernel zone and its non-global zones must run at least Oracle Solaris 11.2. Existing `solaris` zones running Oracle Solaris 11 or Oracle Solaris 11.1 must be updated to at least Oracle Solaris 11.2 before they can run in a kernel zone. See Chapter 3, Installing and Updating Software Packages in *Updating Systems and Adding Software in Oracle Solaris 11.4* for information about updating system software packages.

Network Configuration
A `solaris` zone that runs in a kernel zone must use exclusive-IP networking, so you must configure the kernel zone to allow for additional MAC addresses as shown in How to Add Multiple MAC Addresses to a Kernel Zone.

System Resources
Zones running in the kernel zone can only use system resources that are available to the kernel zone. These resources include virtual disks and iSCSI disks.

Cloning
If you clone a kernel zone that contains non-global zones, only the outside kernel zone will be cloned. The zones inside the kernel zone are not cloned during the zone cloning process. See Cloning a Kernel Zone.

How to Add Multiple MAC Addresses to a Kernel Zone

1. Become a zone administrator.

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Add the new MAC addresses.

   ```
   global$ pfbash zonecfg -z kzone
   zonecfg:kzone> add anet
   zonecfg:kzone:anet> add mac
   zonecfg:kzone:anet:mac> end
   zonecfg:kzone:anet> add mac
   zonecfg:kzone:anet:mac> end
   zonecfg:kzone:anet> end
   zonecfg:kzone> exit
   ```

3. Apply the changes to the running kernel zone.

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

You can alternatively boot the zone to apply the changes.

4. Log in to the kernel zone and verify the new MAC addresses.

```
global$ zlogin kzone
kzone$ pfbash 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    --
```

Configuring New `solaris` Zones in a Kernel Zone

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

```
kzone$ pfbash zonecfg -z new-zone
Use 'create' to begin configuring a new zone.
zonecfg:new-zone> create -t SYSsolaris
zonecfg:new-zone> commit
zonecfg:new-zone> exit
kzone$ zoneadm -z new-zone install
kzone$ zoneadm -z new-zone boot
```

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 Type
- Time of day offset between the kernel zone clock and the global zone clock
- OpenBoot variables (SPARC only)

Storage Information From Host Data

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 enters the `unavailable` state and an error message indicates which system is using the boot storage. For example:

```
global$ pfbash 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.

Encryption Keys and Host Data

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. 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, generate a new encryption key and host data by running the following command:

```
$ pfbash zoneadm -z kernel-zone attach -x initialize-hostdata
```

To boot, a kernel zone must have the correct keysource defined. A migration with `zoneadm migrate` copies the keysource data from the source host along with the zone configuration if a configuration is not already defined on the target host.

If you want to create the zone configuration and keysource on the target host before migration, use the `zonecfg export` command on the source host to export the information to a file that you can use to create the configuration on the target host with the correct keys. For example, to create the configuration for a zone you will migrate from `global1` to `global2` export the configuration on `global1` to a file on a network path and create the configuration on `global2` from that file:

```
global1$ zonecfg -z kzone1 export -f /net/example/path/kzone1.cfg
global2$ zonecfg -z kzone1 -f /net/example/path/kzone1.cfg
```

If the zone's keys for an existing configuration on a target host are not correct, when you try to attach or boot the zone you see the following message:

```
zone 'kzone1': error: Encryption key is incorrect.  See solaris-kz(7) for
configuration migration
zone 'kzone1': procedure or update /etc/zones/keys/kzone1.
```

Keys might be incorrect, for example, if the zone was reinstalled or if the zone was attached with `-x initialize-hostdata`, which reinitializes the keys.

You can fix the problem by deleting the zone configuration on the target host and exporting the configuration again, or by migrating the zone to the target host without creating the configuration first.

Invoking 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 the tilde followed by a dot, `~.`.

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 11.4 Boot Environments*. Additional information for managing zones and boot environments is available in Chapter 2, beadm Zones Support in *Creating and Administering Oracle Solaris 11.4 Boot Environments*.

How to Specify Alternate Boot Environments in a Kernel Zone

1. Become a zone administrator.

   For more information, see Using Rights Profiles to Install and Manage Zones in *Creating and Using Oracle Solaris Zones*.

2. Log in to the kernel zone console.

   ```
   global$ pfbash zlogin -C kzone
   ```

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

   Output is similar to the following:

   ```
   global$ pfbash zoneadm -z kzone boot -- -L
   [Connected to zone 'kzone' 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.

   ```
   global$ zoneadm -z kzone boot -- -Z boot-environment
   ```

Example 7-1 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 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 7-2 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:
```

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.

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 are examples of 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`(8) 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 might 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.

See the `dumpadm`(8) and `coreadm`(8) man pages for additional information.

Index