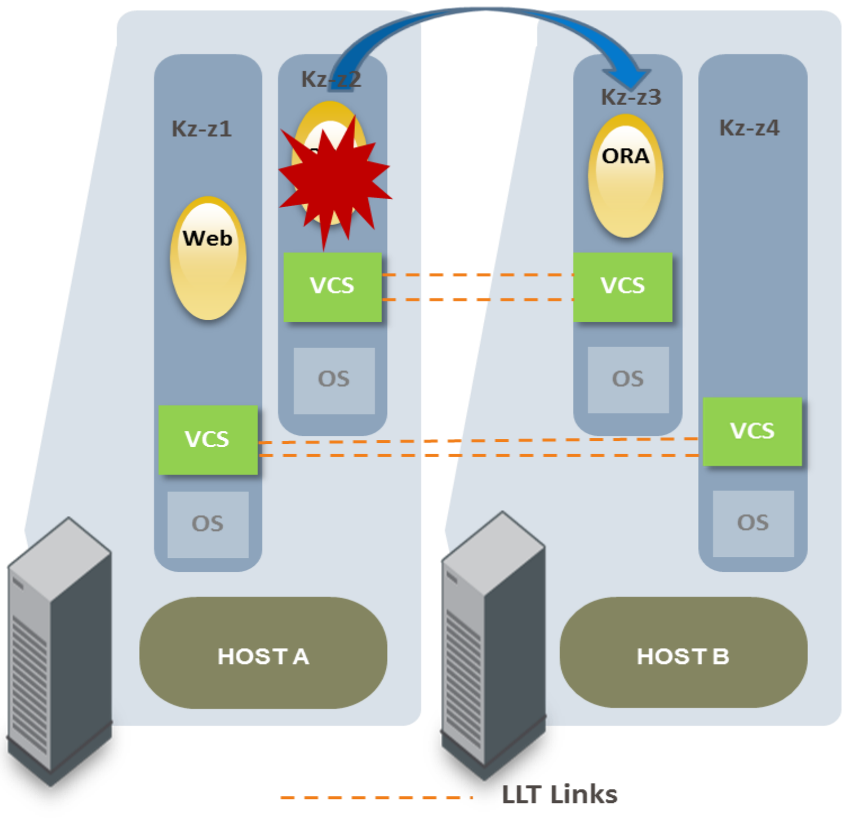
VERITAS Cluster Server support for Oracle Solaris Kernel Zones

Oracle Solaris introduced kernel zones feature in Oracle Solaris 11.2, which provides the flexibility, efficiency and isolation of Oracle Solaris Zones while adding the capability for each kernel zone to run in an independent kernel. Each kernel zone running on a host can be at a different kernel level and be updated individually. This allows application requiring different kernel version requirements to safely run side by side on the same system.

Oracle Solaris kernel zone is supported with VERITAS Cluster Server from version 6.2 onwards. Below mentioned are some of the scenarios in which VERITAS Cluster Server can be used to provide availability to the applications running inside kernel zone and/or to the kernel zone itself.

Kernel zone in-guest clustering

In this scenario failover of the individual application happens from one kernel zone to another much like traditional VCS clusters.



- The entire VCS stack is installed inside the kernel zone including the kernel components like LLT, GAB, and Fencing etc.
- VCS Cluster is formed between kernel zones.
- Multiple kernel zone clusters can be configured between hosts. It is recommended to create kernel zone clusters between zones residing on different hosts. VCS supports kernel zone cluster configured with kernel zones running on the same host as well.
- Fencing in kernel zone cluster supported only through CPS/Majority mode fencing.
- Zpools inside the kernel zone can be managed through VCS cluster.

Example

Below is the zone configuration for the zone Kz-z2 configured on Host A and zone Kz-z3 configured on Host B. The Zone root path is located on a Zpool volume. The network device net9 is used to access the public network whereas net10 and net11 is used for LLT link communication. Majority mode fencing is configured in the kernel zone cluster for arbitration purpose in case of split-brain scenarios. Please refer to the *VCS Administrators guide* for more information on Majority mode fencing.

Step 1: Configuration of kernel zone Kz-z2

```
[root@HostA ~]# zonecfg -z Kz-z2 info
zonename: Kz-z2
brand: solaris-kz
autoboot: false
autoshutdown: shutdown
bootargs:
pool:
scheduling-class:
hostid: 0x2d702515
tenant:
net:
        allowed-address not specified
        configure-allowed-address: true
        physical: net9
        defrouter not specified
        id: 0
net:
        allowed-address not specified
        configure-allowed-address: true
        physical: net10
        defrouter not specified
        id: 1
net:
        allowed-address not specified
        configure-allowed-address: true
        physical: net11
        defrouter not specified
        id: 2
device:
        match not specified
        storage: dev:/dev/zvol/dsk/Kz-z2-pool/kz2
        id: 1
        bootpri: 0
device:
        match: /dev/rdsk/c1d4s2
        storage not specified
        id: 2
        bootpri not specified
capped-memory:
        physical: 6G
keysource:
        raw redacted
```

Step 2: Configuration of kernel zone Kz-z3

```
[root@HostB ~]#zonecfg -z Kz-z3 info
zonename: Kz-z3
brand: solaris-kz
autoboot: false
autoshutdown: shutdown
bootargs:
pool:
scheduling-class:
hostid: 0x1729bed5
tenant:
net:
        allowed-address not specified
        configure-allowed-address: true
        physical: net9
        defrouter not specified
        id: 0
net:
        allowed-address not specified
        configure-allowed-address: true
        physical: net11
        defrouter not specified
        id: 2
net:
        allowed-address not specified
        configure-allowed-address: true
        physical: net10
        defrouter not specified
        id: 1
```

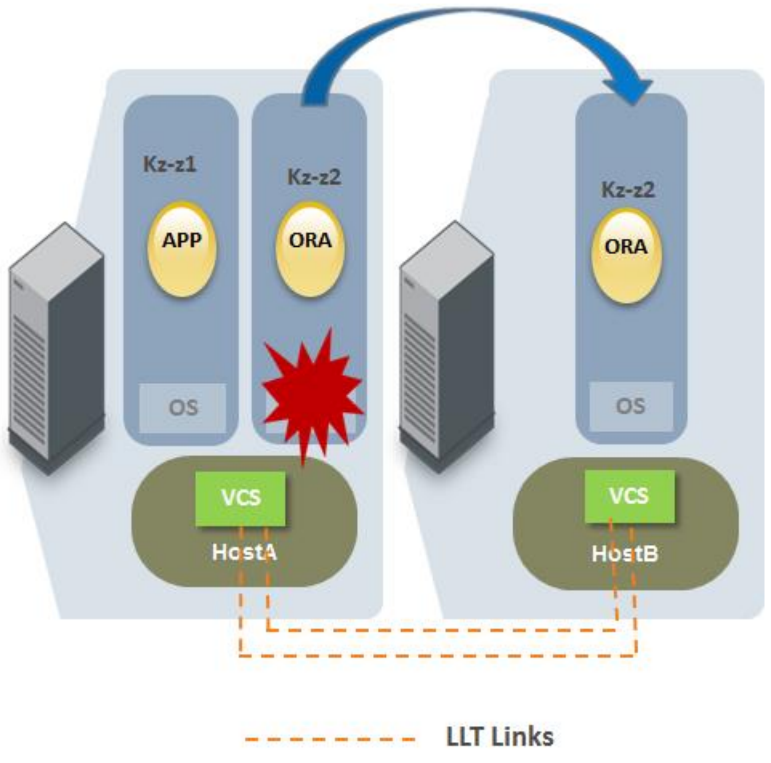
```
device:
        match not specified
        storage: dev:/dev/zvol/dsk/Kz-z3-pool/kz3
        id: 1
        bootpri: 0
device:
        match: /dev/rdsk/c1d4s2
        storage not specified
        id: 2
        bootpri not specified
capped-memory:
        physical: 6G
keysource:
        raw redacted
```

Step 3: The main.cf configuration file for the cluster inside the kernel zones

```
include "OracleASMTypes.cf"
include "types.cf"
include "OracleTypes.cf"

cluster kzzclus (
        SecureClus = 1
        UseFence = SCSI3
        )

system Kz-z2 (
        )

system Kz-z3 (
        )

group kzorasg (
        SystemList = { Kz-z2 = 0, Kz-z3 = 1 }
        )

        IP kzip (
                Device = net0
                Address = "192.168.10.54"
                NetMask = "255.255.255.0"
                )

        NIC kznic (
                Device = net0
                NetworkHosts = { "192.168.5.1" }
                )

        Netlsnr kzlsnr (
                Owner = oracle
                Home = "/u01/app/oracle/product/11.2.0/dbhome_1"
                Listener = KZLSNR
                )

        Oracle kzoradb (
                Sid = kzdb
                Owner = oracle
                Home = "/u01/app/oracle/product/11.2.0/dbhome_1"
                User = vcsuser
                Pword = ftlRitItkTirKth
                Table = vcs
                )

        Zpool kzorapool (
                PoolName = kz-orapool
                AltRootPath = "/"
                )

        kzip requires kznic
        kzlsnr requires kzip
        kzlsnr requires kzoradb
        kzoradb requires kzorapool
```

Kernel Zone high availability

In this scenario kernel zone is failed over from one host to another. Zone agent is used to monitor the kernel zone from the global zone.



- Zone agent configured in the global zone manages the kernel zone as a resource.
- The kernel zone is configured as part of a failover service group. When the kernel zone faults, VCS ensures that the kernel zone is made available on a different node in the cluster.
- AMF support for kernel zone is available for instant fault detection.
- Applications running inside kernel zone cannot be managed in this scenario.

Example

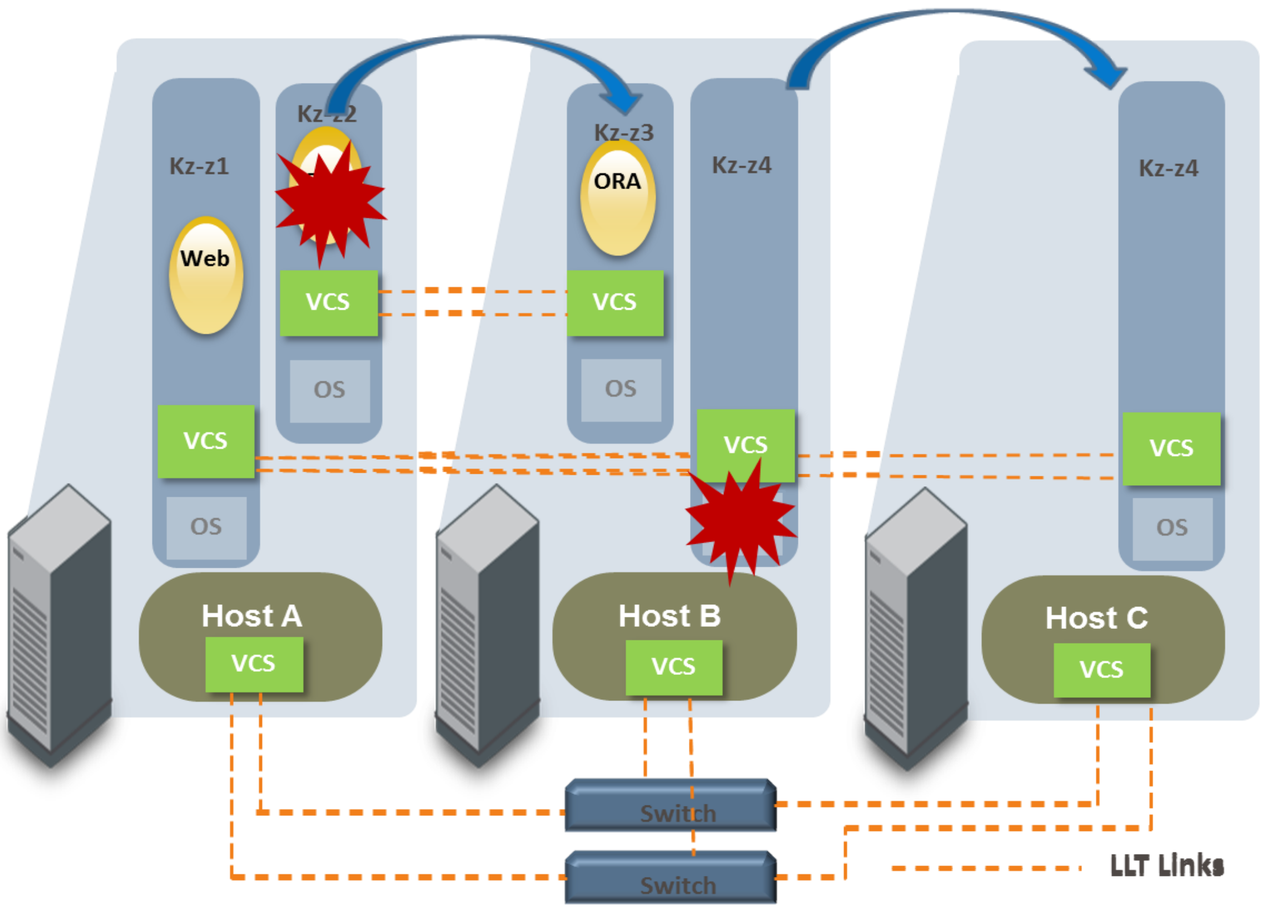
In the above scenario, shared storage has been assigned to the kernel zone for storing oracle database. The oracle database is stored on a Zpool file system.

Step 1: The main.cf configuration file for the cluster on the Physical hosts

```
include "types.cf"

cluster hostclus (
        UserNames = { "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" = 0 }
                SecureClus = 1
        )

system HostA (
        )
system HostB (
        )
group kzonegrp-Kz1 (
        SystemList = { HostA = 0, HostB = 1 }
        ContainerInfo = { Name = Kz-z1, Type = Zone, Enabled = 1 }
         AutoStartList = { HostA, HostB }
        Administrators = { "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" }
        )

        Zone kzres-Kz1 (
                ForceAttach = 0
                )

        Zpool kzpool-Kz1 (
                PoolName = Kz-z1-pool
                AltRootPath = "/"
                ChkZFSMounts = 0
                )
        Kzres-Kz1 requires kzpool-Kz1

group kzonegrp-Kz2 (
        SystemList = { HostA = 0, HostB = 1 }
        ContainerInfo = { Name = Kz-z2, Type = Zone, Enabled = 1 }
         AutoStartList = { HostA, HostB }
        Administrators = { "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" }
        )

        Zone kzres-Kz2 (
                ForceAttach = 0
                )

        Zpool kzpool-Kz2 (
                PoolName = Kz-z2-pool
                AltRootPath = "/"
                ChkZFSMounts = 0
                )

        Kzres-Kz2 requires kzpool-Kz2
```

Kernel Zone high availability with Application failover

In this scenario, VCS is installed both in global zone and kernel zones. VCS installed on the global zone manages the availability of Kernel Zone and the VCS installed inside the kernel zones manage the application availability within the kernel zones.



- In the above environment, cluster server ensures that both the application running inside the kernel and the infrastructure i.e. kernel zone is made available in case of a failure in any of these components.
- It is recommended to use separate private heart beat links for the cluster server running on the host and the cluster server running inside the kernel zone guest.

Limitations: Administrative reboot inside the zone may be treated as a zone fault by the zone agent and may failover the zone to another node. To avoid accidental failover of the service group you need to either set the “ToleranceLimit” of the kernel zone resource to a higher value (>1) or freeze the service group or set the kernel zone resource “Critical” attribute to 0.

Example

For the kernel zone cluster configuration refer to section *kernel zone in-guest clustering.*

Step 1: The main.cf configuration file configured on the Physical host cluster

```
include "types.cf"

cluster hostclus (
        UserNames = {
                "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" = 0 }
                SecureClus = 1
        )

system HostA (
        )
system HostB (
        )
system HostC (
        )

group kzonegrp-Kz1 (
        SystemList = { HostA = 0, HostC = 1, HostB = 2 }
        ContainerInfo = { Name = Kz-z1, Type = Zone, Enabled = 1 }
         AutoStartList = { HostA, HostC,HostB }
        Administrators = { "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" }
        )

        Zone kzres-Kz1 (
                ForceAttach = 0
                )

        Zpool kzpool-Kz1 (
                PoolName = Kz-z1-pool
                AltRootPath = "/"
                ChkZFSMounts = 0
                )

        Kzres-Kz1 requires kzpool-Kz1

group kzonegrp-Kz2 (
        SystemList = { HostA = 0, HostC = 1, HostB = 2 }
        ContainerInfo = { Name = Kz-z2, Type = Zone, Enabled = 1 }
         AutoStartList = { HostA}
        Administrators = { "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" }
        )

        Zone kzres-Kz2 (
                ForceAttach = 0
```

```
		)

	Zpool kzpool-Kz2 (
		PoolName = Kz-z2-pool
		AltRootPath = "/"
		ChkZFSMounts = 0
		)

	Kzres-Kz2 requires kzpool-Kz2

group kzonegrp-Kz3 (
	SystemList = { HostB = 0, HostC = 1, HostA = 2 }
	ContainerInfo = { Name = Kz-z3, Type = Zone, Enabled = 1 }
	 AutoStartList = { HostB, HostC, HostA }
	Administrators = { "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" }
	)

	Zone kzres-Kz3 (
		ForceAttach = 0
		)

	Zpool kzpool (
		PoolName = Kz-z3-pool
		AltRootPath = "/"
		ChkZFSMounts = 0
		)

	Kzres-Kz3 requires kzpool-Kz3

group kzonegrp-Kz4 (
	SystemList = { HostB = 0, HostC = 1, HostA = 2 }
	ContainerInfo = { Name = Kz-z4, Type = Zone, Enabled = 1 }
	 AutoStartList = { HostB, HostC, HostA }
	Administrators = { "kzusr@vcs_lzs@03f6d8b0-1dd2-11b2-8fba-00144ff9d281" }
	)

	Zone kzres-Kz4 (
		ForceAttach = 0
		)

	Zpool kzpool-Kz4 (
		PoolName = Kz-z4-pool
		AltRootPath = "/"
		ChkZFSMounts = 0
		)

	Kzres-Kz4 requires kzpool-Kz4
```