



Implementing IBM DB2 UDB V9 HA in a Solaris™ Cluster 3.2 Environment

Cherry Shu and Neil Garthwaite

December 2007

Sun Microsystems, Inc.

Copyright © 2007 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms. This distribution may include materials developed by third parties.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and in other countries, exclusively licensed through X/Open Company, Ltd. X/Open is a registered trademark of X/Open Company, Ltd.

All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the United States and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

Sun, Sun Microsystems, the Sun logo, Java, OpenSolaris, Solaris, and Sun Cluster are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

This product is covered and controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Table of Contents

Introduction.....	4
IBM DB2 UDB HA Agent for Sun Cluster 3.x Software.....	4
Configuration Summary and Prerequisites.....	4
Implementation Checklist.....	5
Install and Configure Solaris Cluster 3.2 Software.....	5
Create ZFS File System.....	6
Install DB2 Binary and Create DB2 Instance.....	7
Enable DB2 Instance for High Availability on ZFS File System.....	9
Test Cluster Failover on ZFS File System.....	12
Enable DB2 Instance for High Availability in Solaris Containers.....	14
Test Cluster Failover Within Solaris Containers.....	18
For More Information.....	19

Introduction

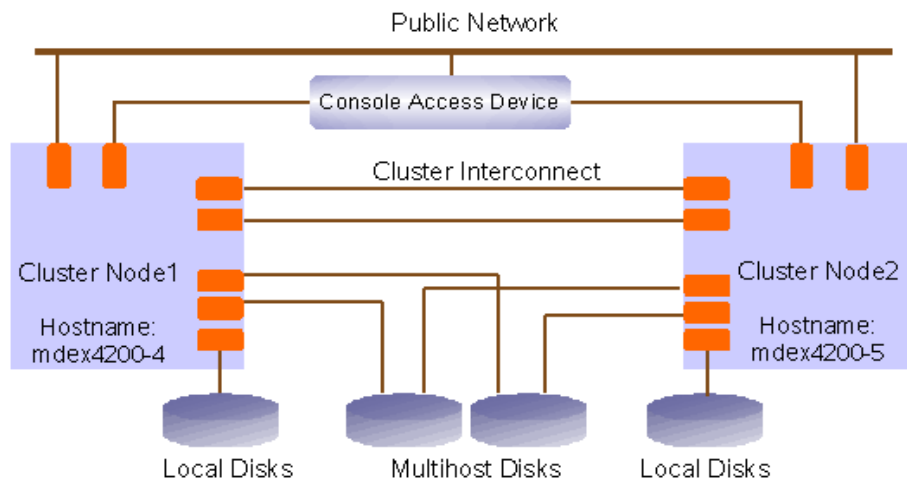
This paper provides step-by-step instructions on how to install, create, and enable IBM DB2 Universal Database (UDB) V9 for high availability (HA) in a two-node Solaris Cluster 3.2 environment. This article demonstrates how to use ZFS as a failover file system for a DB2 instance and how to implement DB2 failover across Solaris Containers in the Solaris 10 Operating System, releases 11/06 and above. *(Please note: As of release 3.2, "Sun Cluster" was renamed "Solaris Cluster".)*

IBM DB2 UDB HA Agent for Sun Cluster 3.x Software

The DB2 UDB HA agent integrates the robust, highly available Sun clustering technology with DB2 UDB database technology. The HA agent provides the methods required to start, stop, monitor, and administer DB2 UDB in a Sun Cluster 3.x environment. The HA agent is developed by IBM and jointly certified by IBM and Sun Microsystems. The agent is part of the DB2 software distribution, so no additional download is necessary.

Configuration Summary and Prerequisites

The two-node cluster is set up as shown in the following figure.



Ensure that the following prerequisites are met on both systems before the installation:

- Hardware is physically configured. The cluster hardware is a supported configuration for the Solaris Cluster 3.2 software.
- Each cluster node has two network interfaces attached to the public network.
- Each cluster node has two additional network interfaces to be used as private interconnects. This example uses two cross-over cables for private interconnects.
- Shared storage is connected to the two nodes.
- Each node has a dedicated internal disk and slice mounted as `/globaldevices`.
- The Solaris 10 11/06 OS is installed. Solaris Cluster 3.2 software supports only the Solaris 10 11/06 release and above.
- The latest Solaris patches have been installed.

Implementation Checklist

To simplify the DB2 UDB V9 HA implementation in a Solaris Cluster 3.2 environment, the implementation is staged in the following seven steps:

1. Install and configure Solaris Cluster software.
2. Create a ZFS file system.
3. Install DB2 binary and create DB2 instance.
4. Enable DB2 instance for high availability on the ZFS file system.
5. Test cluster failover on the ZFS file system.
6. Enable DB2 instance for high availability in Solaris Containers.
7. Test cluster failover within Solaris Containers.

The remaining sections of this document provide detailed instructions for the implementation. The instructions are valid on both the Solaris OS for x64 platforms and the Solaris OS for SPARC® platforms.

Install and Configure Solaris Cluster 3.2 Software

1. Install the Solaris Cluster software packages.

On each node, run the Solaris Cluster installer program. For more information about the installer program, see the *Sun Cluster Software Installation Guide for Solaris OS*.

Follow the on-screen instructions to install the Solaris Cluster framework package. Select the following components to install:

```
Java DB
Sun Cluster 3.2
All Shared Components
```

Select `Configure Later` on the Configuration Type Panel.

Set up the root `.profile` to add cluster commands to the `$PATH` and `$MANPATH`. Add the following lines to `.profile`:

```
PATH=/usr/openwin/bin:/usr/cluster/bin:$PATH
MANPATH=/usr/cluster/man:$MANPATH
export PATH MANPATH
```

Ensure “secure by default” networking is off.

```
# svcprop network/rpc/bind:default | grep local_only (Should be false)
# svcprop /system/webconsole:console | grep tcp_listen (Should be true)
```

2. Configure the Solaris Cluster software.

Log on to one of the two nodes. Start the `scinstall` utility in interactive mode as root user:

```
# ./profile
# scinstall
```

Select the first node of a cluster, and then select the typical cluster install. Follow the menu prompts to supply the following information:

- Name to give the cluster: `db2cluster`
- Name of the other node (if started from node `mdex4200-4`): `mdex4200-5`
- The first private adapter name (interface name): `e1000g1`
- The second private adapter name (interface name): `e1000g2`
- Type `no` to disable automatic quorum device selection
- Type `no` to interrupt cluster creation for `sccheck` errors
- Allow auto reboot

The installer program configures the cluster and reboots the node. After it reboots, try `cluster status`. If that works, continue.

On the second node, run the `scinstall` utility. Select `add a cluster node`. Follow the menu prompts to supply the following information:

- Name to give the cluster: `db2cluster`
- Enter the sponsoring node name: `mdex4200-4`
- Accept the interconnects found
- Allow auto reboot

The cluster is established when both nodes have successfully booted into the cluster. The installation logs are in the `/var/cluster/logs/install` directories.

3. Verify the nodes and the quorum device are successfully configured.

On one of the nodes, use `clquorum list` to check whether the cluster is successfully established:

```
# clquorum list
d5
mdex4200-4
mdex4200-5
```

Create ZFS File System

Solaris Cluster 3.2 software provides support for ZFS as a failover file system, and the ZFS pool is the unit of failover. The `HAStoragePlus` resource type controls the failover of ZFS file systems contained in one or more pools.

Please note that if ZFS is used as a failover file system, it is not possible to deploy multiple database partitions across multiple machines (also known as Massively Parallel Processing, or MPP). This configuration requires a cluster file system. Instead, single partition and logical database partitions configurations (also known as Multiple Logical Nodes, or MLN) are supported with ZFS.

In the following example, a ZFS pool is created using two mirrored disks. For more robust redundancy, you can create a RAIDZ pool, which is ZFS-specific, RAID-5-like redundancy. With ZFS, you can completely skip any traditional cluster device group management. `HAStoragePlus` provides failover of an entire pool, thus replacing the role normally taken by device groups.

Create a ZFS pool named `db2hapool` contained in a mirror of two disks. Note that the disks are accessible from both nodes, and traditional disk paths are used instead of DID devices.

```
node1# zpool create db2hapool mirror \  
/dev/dsk/c6t600C0FF00000000008565B7A38657A00d0\  
/dev/rdisk/c6t600C0FF000000000004B70331CD4C700d0  
  
node1# zpool status  
pool: db2hapool  
state: ONLINE  
scrub: none requested  
config:  
NAME STATE READ WRITE CKSUM  
db2hapool ONLINE 0 0 0  
mirror ONLINE 0 0 0  
c6t600C0FF00000000008565B7A38657A00d0 ONLINE 0 0 0  
c6t600C0FF000000000004B70331CD4C700d0 ONLINE 0 0 0  
errors: No known data errors
```

Now create file systems in the pool. ZFS automatically mounts the file systems.

```
node1# zfs create db2hapool/db2zfs1  
node1# zfs create db2hapool/db2zfsf1  
node1# zfs create db2hapool/db2zfsas  
  
node1# zfs list  
NAME                               USED  AVAIL  REFER  MOUNTPOINT  
db2hapool                          168K  134G   28.5K  /db2hapool  
db2hapool/db2zfs1                   24.5K  134G   24.5K  /db2hapool/db2zfs1  
db2hapool/db2zfsas                  24.5K  134G   24.5K  /db2hapool/db2zfsas  
db2hapool/db2zfsf1                  24.5K  134G   24.5K  /db2hapool/db2zfsf1
```

Install DB2 Binary and Create DB2 Instance

The DB2 UDB binary will be installed on a cluster file system. To simplify the configuration, the same cluster file system is also used as the instance home directory mount point.

1. Create a cluster file system.

In this example, we utilized the built-in Solaris Volume Manager with the Solaris 10 OS to create disk sets. Here is the sample `/etc/vfstab` for mounting the cluster file system on both nodes:

```
# DB2 global devices  
/dev/md/dg_d1/dsk/d100 /dev/md/dg_d1/rdsk/d100 /global/opt/IBM ufs 3 yes global, logging
```

Create a link to `/opt/IBM`:

```
# ln -s /global/opt/IBM /opt/IBM
```

2. Install DB2 UDB V9 on the cluster file system.

On one of the nodes, log in as root user, run `db2_install` and choose ESE to install the complete binary set. Take the default `/opt` as the base directory. The DB2 binary will be installed in `/opt/IBM/db2/V9.1`.

3. Create user and group accounts.

On the same node, create three separate groups and user accounts for the DB2 instance owner, the DB2 fence user, and the administration server. DB2 users use the ZFS file systems created in the previous section as the home directories.

```
# groupadd -g 302 zdb2iadm
# groupadd -g 301 zdb2fadm
# groupadd -g 300 zdb2as
# useradd -g zdb2iadm -u 412 -d /db2hapool/db2zfs1 -m -s /bin/ksh db2zfs1
# useradd -g zdb2fadm -u 411 -d /db2hapool/db2zfsf1 -m -s /bin/ksh db2zfsf1
# useradd -g zdb2as -u 410 -d /db2hapool/db2zfsas -m -s /bin/ksh db2zfsas
```

Then set the account passwords for the three users using `passwd(1M)`.

4. Create a DB2 instance.

```
# /global/opt/IBM/db2/V9.1/instance/db2icrt -a server -u db2zfsf1 db2zfs1
```

5. Update `svcname` and `/etc/services` for the DB2 instance.

```
# su - db2zfs1 -c "db2 update dbm cfg using svcname xdb2zfs1"
```

Update `/etc/services` to reflect the port definition for client connections and the reserved FCM ports by adding the following lines to `/etc/services`:

```
DB2_db2zfs1      60012/tcp
DB2_db2zfs1_1   60013/tcp
DB2_db2zfs1_2   60014/tcp
DB2_db2zfs1_END 60015/tcp
xdb2zfs1        60016/tcp
```

6. Set the IPC resource controls.

DB2 provides a utility called `db2osconf` to generate the recommended values for IPC tunables on the Solaris platform:

```
# /global/opt/IBM/db2/V9.1/bin/db2osconf
set msgsys:msginfo_msgmni = 6144
set semsys:seminfo_semmni = 7168
set shmsys:shminfo_shmmax = 9578697523
set shmsys:shminfo_shmmni = 7168
```

```
Total kernel space for IPC:
0.98MB (shm) + 1.71MB (sem) + 2.08MB (msg) == 4.77MB (total)
```

Set the IPC tunables via resource controls for DB2 instance owner `db2zfs1`. (Backslashes are used here to show that commands are split over multiple lines.)

```
# projadd -c "DB2 instance on ZFS" user.db2zfs1
# projmod -s -K \
  "project.max-shm-memory=(privileged,9578697523,deny)" \
  user.db2zfs1
# projmod -s -K "project.max-shm-ids=(privileged,7168,deny)" \
  user.db2zfs1
# projmod -s -K "project.max-msg-ids=(privileged,6144,deny)" \
  user.db2zfs1
# projmod -s -K "project.max-sem-ids=(privileged,7168,deny)" \
  user.db2zfs1
```

7. Test starting and stopping DB2.

Log in as the instance owner `db2zfs1`:

```
# su - db2zfs1
```

To enable user remote shell access from any node in the cluster, create an `.rhosts` file containing the following line:

```
+ db2zfs1
```

Run `db2start` and `db2stop` to see if DB2 can start and stop successfully.

8. Export the ZFS pool.

```
# zpool export db2hapool
```

9. Import the ZFS pool on the other node.

```
# zpool import db2hapool
```

10. To configure the DB2 environment on the other node, repeat steps 3, 5, and 6 on the other node.

11. Manually test starting and stopping DB2 on the other node.

Log in as the instance owner `db2zfs1`, and modify `sqllib/db2nodes.cfg` to change the host name, for example:

```
0 mdex4200-5 0
```

Run `db2start` and `db2stop` to see if DB2 can start and stop successfully.

12. Get the ZFS pool exported and imported back on to cluster node 1 (`mdex4200-4`), and modify `sqllib/db2nodes.cfg` to change the host name to cluster node 1.

Enable DB2 Instance for High Availability on ZFS File System

DB2 provides four scripts to control the registration, unregistration, onlining, and offlining of DB2 UDB in a Solaris Cluster 3.x environment. Those scripts are under the `/opt/IBM/db2/V9.1/ha/sc30/util` directory. Note that these scripts use the old command-line interface of the Sun Cluster 3.1 and 3.0 software to perform the tasks. The old commands continue to exist in the Solaris Cluster 3.2 software. In this paper, we use the new command set of the Solaris Cluster 3.2 software, which is designed to be object-oriented rather than task-related.

1. On both nodes, add an entry in `/etc/hosts` and `/etc/inet/ipnodes` for the logical host named `db2zfs1-1h`. The example `/etc/hosts` or `/etc/inet/ipnodes` on `mdex4200-4` looks like the following.

```

127.0.0.1      localhost
10.8.2.125    mdex4200-4  loghost mdex4200-4.
10.8.2.124    mdex4200-5

10.8.2.130    db2zfs1-1h

172.16.4.2    clprivnet0

```

2. Disable entries in `/etc/inittab` that automatically start DB2 processes by commenting them out with a leading `#`, for example:

```
#fmc:234:respawn:/global/opt/IBM/db2/V9.1/bin/db2fmcd #DB2 Fault Monitor Coordinator
```

3. Copy the `$DB2INSTALLPATH/ha/sc30` directory to the `$INSTANCEHOME/sqllib` directory.

```
$ cp -pr /global/opt/IBM/db2/V9.1/ha /db2hapool/db2zfs1/sqllib
```

4. Register the Solaris Cluster resource type for the DB2 HA agent.

DB2 provides the `regdb2udb` script to control the registration of DB2 UDB in a Solaris Cluster environment. But you cannot use it directly in this case because the DB2 instance home is on a ZFS file system, which is not a HA file system yet.

Create a Resource Type Registration (RTR) file for the DB2 instance `db2zfs` by customizing the template RTR file `IBM.db2udb`:

```
# sed -e "s|RT_BASEDIR.*|RT_BASEDIR=/opt/IBM/db2/V9.1/ha/sc30/bin ;|g" \
-e "s|RESOURCE_TYPE.*|RESOURCE_TYPE=db2udb_db2zfs1;|g" \
/opt/IBM/db2/V9.1/ha/sc30/etc/IBM.db2udb
> /opt/IBM/db2/V9.1/ha/sc30/etc/IBM.db2udb.db2zfs1
```

Copy the RTR file to `/usr/cluster/lib/rgm/rtreg` on both nodes:

```
# cd /opt/IBM/db2/V9.1/ha/sc30/etc
# cp IBM.db2udb.db2zfs1 /usr/cluster/lib/rgm/rtreg
# rcp IBM.db2udb.db2zfs1 mdex4200-5:/usr/cluster/lib/rgm/rtreg
```

Register the resource type:

```
# clresourcetype register IBM.db2udb_db2zfs1
# clresourcetype show IBM.db2udb_db2zfs1
```

```
=== Registered Resource Types ===
```

```
Resource Type:          IBM.db2udb_db2zfs1
RT_description:         DB2 UDB Resource Type on Sun Cluster 3.0
RT_version:             1.0
API_version:           2
RT_basedir:            /opt/IBM/db2/V9.1/ha/sc30/bin
Single_instance:       False
Proxy:                 False
Init_nodes:            All potential masters
Installed_nodes:       <All>
Failover:              False
Pkglist:               IBMdb2udb
RT_system:             False
Global_zone:           False
```

5. Create a resource group for the DB2 instance:

```
# clresourcegroup create db2_db2zfs1_0-rg
```

6. Create a logical host resource for the DB2 instance:

```
# clreslogicalhostname create -g db2_db2zfs1_0-rg -h db2zfs1-lh db2zfs1-lh
```

7. Create a DB2 failover resource with a resource of type SUNW.HASStoragePlus and configure the ZFS pool db2hapool to the resource:

```
# clresource create -g db2_db2zfs1_0-rg -t SUNW.HASStoragePlus -p zpools=db2hapool
db2zfs1-has
```

8. Enable the DB2 resource group:

```
# clresourcegroup online -M db2_db2zfs1_0-rg
```

9. Set the DB2 global register variable and database manager configuration parameter for failover control:

```
# su - db2zfs1 -c "db2set DB2_NUM_FAILOVER_NODES=1"
# su - db2zfs1 -c "db2 update dbm cfg using START_STOP_TIME 2"
```

10. Create the DB2 HA resource:

```
# clresource create -g db2_db2zfs1_0-rg \
-t IBM.db2udb_db2zfs1 \
-p INSTHOME=/db2hapool/db2zfs1 \
-p DB2INSTANCE=db2zfs1 \
-p DB2_NODE_NUMBER=0 \
-p Resource_dependencies=db2zfs1-lh,db2zfs1-has\
db2_db2zfs1_0-rs
```

11. Verify the resource group and resources status.

```
# clresourcegroup show
Resource Group: db2_db2zfs1_0-rg
RG_description: <NULL>
RG_mode: Failover
RG_state: Managed
Failback: False
Nodelist: mdex4200-4 mdex4200-5

--- Resources for Group db2_db2zfs1_0-rg ---

Resource: db2zfs1-lh
Type: SUNW.LogicalHostname:2
Type_version: 2
Group: db2_db2zfs1_0-rg
R_description:
Resource_project_name: default
Enabled{mdex4200-4}: False
Enabled{mdex4200-5}: False
Monitored{mdex4200-4}: True
Monitored{mdex4200-5}: True
```

```

Resource:                               db2zfs1-has
Type:                                   SUNW.HAStoragePlus:4
Type_version:                           4
Group:                                   db2_db2zfs1_0-rg
R_description:                           default
Resource_project_name:                   default
Enabled{mdex4200-4}:                     True
Enabled{mdex4200-5}:                     True
Monitored{mdex4200-4}:                   True
Monitored{mdex4200-5}:                   True

Resource:                               db2_db2zfs1_0-rs
Type:                                   IBM.db2udb_db2zfs1
Type_version:                             1.0
Group:                                   db2_db2zfs1_0-rg
R_description:                           default
Resource_project_name:                   default
Enabled{mdex4200-4}:                     False
Enabled{mdex4200-5}:                     False
Monitored{mdex4200-4}:                   True
Monitored{mdex4200-5}:                   True

```

That's all there is to it. The DB2 instance `db2zfs1` deployed on the ZFS file system will now fail over between the two nodes.

Test Cluster Failover on ZFS File System

Take all the resources offline and then bring them all online using the following Solaris Cluster commands:

```

# clresource disable db2_db2zfs1_0-rs
# clresource disable db2zfs1-has
# clresource disable db2zfs1-lh
# clresource enable db2zfs1-lh
# clresource enable db2zfs1-has
# clresource enable db2_db2zfs1_0-rs

```

Assume at this point that the resource group `db2_db2zfs1_0-rg` is online at `mdex4200-4`. Then test the failover of the DB2 instance and associated resources from `mdex4200-4` to `mdex4200-5`.

```

# clresourcegroup switch -n mdex4200-5 db2_db2zfs1_0-rg

```

You should now see that the DB2 resources for `db2zfs1` are hosted by the secondary node `mdex4200-5`. Verify this by executing the `cluster status` command.

```

# cluster status

=== Cluster Nodes ===

--- Node Status ---

Node Name                               Status
-----
mdex4200-4                               Online
mdex4200-5                               Online

=== Cluster Transport Paths ===

Endpoint1                               Endpoint2                               Status
-----
mdex4200-4:e1000g2                     mdex4200-5:e1000g2                     Path online

```

mdex4200-4:e1000g1 mdex4200-5:e1000g1 Path online

=== Cluster Quorum ===

--- Quorum Votes Summary ---

Needed	Present	Possible
2	3	3

--- Quorum Votes by Node ---

Node Name	Present	Possible	Status
mdex4200-4	1	1	Online
mdex4200-5	1	1	Online

--- Quorum Votes by Device ---

Device Name	Present	Possible	Status
d5	1	1	Online

=== Cluster Device Groups ===

--- Device Group Status ---

Device Group Name	Primary	Secondary	Status
dg_d1	mdex4200-5	mdex4200-4	Online

--- Spare, Inactive, and In Transition Nodes ---

Device Group Name	Spare Nodes	Inactive Nodes	In Transition Nodes
dg_d1	-	-	-

--- Multi-owner Device Group Status ---

Device Group Name	Node Name	Status
-------------------	-----------	--------

=== Cluster Resource Groups ===

Group Name	Node Name	Suspended	State
db2_db2zfs1_0-rg	mdex4200-4	No	Offline
	mdex4200-5	No	Online

=== Cluster Resources ===

Resource Name	Node Name	State	Status Message
db2zfs1-lh	mdex4200-4	Offline	Offline - LogicalHostname offline.
	mdex4200-5	Online	Online - LogicalHostname online.
db2zfs1-has	mdex4200-4	Offline	Offline
	mdex4200-5	Online	Online
db2_db2zfs1_0-rs	mdex4200-4	Offline	Offline
	mdex4200-5	Online	Online

=== Cluster DID Devices ===

Device Instance	Node	Status
/dev/did/rdisk/d1	mdex4200-4	Ok
/dev/did/rdisk/d10	mdex4200-4 mdex4200-5	Ok Ok
/dev/did/rdisk/d11	mdex4200-5	Ok
/dev/did/rdisk/d12	mdex4200-5	Ok
/dev/did/rdisk/d2	mdex4200-4	Ok
/dev/did/rdisk/d5	mdex4200-4 mdex4200-5	Ok Ok
/dev/did/rdisk/d6	mdex4200-4 mdex4200-5	Ok Ok
/dev/did/rdisk/d7	mdex4200-4 mdex4200-5	Ok Ok
/dev/did/rdisk/d8	mdex4200-4 mdex4200-5	Ok Ok
/dev/did/rdisk/d9	mdex4200-4 mdex4200-5	Ok Ok

You can simulate a power-off test by killing all running `db2sysc` processes to verify the cluster failover capabilities.

To take the DB2 instance offline, you must take offline the resource group as follows.

```
#clresource disable -g db2zfs1_0_rg +
```

To remove the DB2 instance from Solaris Cluster 3.2 software monitoring and control, remove the resources as follows.

```
#clresource delete -g db2zfs1_0_rg +  
#clresourcegroup delete db2zfs1_0_rg
```

Enable DB2 Instance for High Availability in Solaris Containers

A new feature of Solaris Cluster software is resource group manager support for Solaris Containers in the Solaris 10 OS. It provides the ability to run clustered applications in Solaris 10 non-global zones, using the exact same data service agents that control the applications when they run in the global zone.

In this section, we create a failover DB2 cluster configuration running in a pair of non-global zones, one per node.

1. Create a whole root non-global zone named `zone1` on both nodes.

```
# mkdir /zones
# zonecfg -z zone1
zone1: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zone1> create -b
zonecfg:zone1> set zonepath=/zones/zone1
zonecfg:zone1> set autoboot=true
zonecfg:zone1> exit
```

Install the zone:

```
# zoneadm -z zone1 install
```

Boot the zone:

```
# zoneadm -z zone1 boot
```

For the first boot of a zone after installation, the standard system identification questions (such as the naming schemes, locale or time zone, or root password) must be answered by accessing the zone's console. This should be done using the `zlogin(1M)` command.

```
global# zlogin -C db2-zone1
```

After setting the system identification and root password, the zone is ready for use.

2. Create the groups and users for DB2 in `zone1` on both physical nodes.

```
# zlogin zone1
# groupadd -g 402 zdb2iadm
# groupadd -g 401 zdb2fadm
# groupadd -g 300 db2as

# useradd -g zdb2iadm -u 402 -d /db2pool/zdb2test -s /bin/ksh zdb2test
# useradd -g zdb2fadm -u 401 -d /db2pool/zdb2fenc -s /bin/ksh zdb2fenc
# useradd -g db2as -u 300 -d /global/opt/IBM/db2as -s /bin/ksh db2as
```

Then set the account passwords for the three users using `passwd(1M)`.

3. In both non-global zones, create a symbolic link to `/opt/IBM`.

```
# ln -s /global/opt/IBM /opt/IBM
```

4. On one node, e.g., `mdex4200-4`, create and configure a DB2 instance `zdb2test` in `zone1`.

```
# /opt/IBM/db2/V9.1/instance/db2icrt -a server -u zdb2fenc zdb2test
```

Then set the IPC resource controls and add the service entries to `/etc/services` for DB2 instance `zdb2test` in `zone1`.

5. Add a logical host entry to `/etc/hosts` and `/etc/inet/ipnodes` for both non-global zones.

```
#
# Internet host table
#
127.0.0.1      localhost      loghost zone1
::1          localhost      loghost zone1
10.8.2.129    zdb2test-lh
```

6. Register the Solaris Cluster resource type for the DB2 HA agent. This can be done only from the global zone. Creating resource groups is also not allowed from a non-global zone. The `regdb2udb` script doesn't work in this case for the same reason.

Follow the instructions in the previous section to create a Resource Type Registration (RTR) file for the DB2 instance `zdb2test` and copy the RTR file to `/usr/cluster/lib/rgm/rtreg` on both nodes, and then register the resource type in the global zone of one node.

```
# clresourcetype register IBM.db2udb_zdb2test
# clresourcetype show IBM.db2udb_zdb2test

=== Registered Resource Types ===

Resource Type:          IBM.db2udb_zdb2test
RT_description:         DB2 UDB Resource Type on Sun Cluster 3.0
RT_version:             1.0
API_version:            2
RT_basedir:             /zones/zone1/root/db2hapool/zdb2test/sqllib/ha/sc30/bin
Single_instance:       False
Proxy:                  False
Init_nodes:             All potential masters
Installed_nodes:        <All>
Failover:               False
Pkglist:                IBMdb2udb
RT_system:              False
Global_zone:           False
```

7. Create a resource group for the DB2 instance from the global zone, and specify the zone name for both nodes.

```
# clresourcegroup create \
-n mdex4200-4:zone1,mdex4200-5:zone1 \
db2_zdb2test_0-rg
```

8. Create a logical host resource for the DB2 instance from the global zone:

```
# clreslogicalhostname create \
-g db2_zdb2test_0-rg \
-h zdb2test-lh zdb2test-lh
```

9. Create a DB2 failover resource with a resource of type `SUNW.HASStoragePlus`. The `HASStoragePlus` resource manages the ZFS pool `db2hapool`. The ZFS file systems are exposed only in the non-global zone. The `HASStoragePlus` resource ensures `db2hapool` and the cluster file system get imported and the loopback file system (LOFS) is mounted in the non-global zone.

```
# clresource create \
-g db2_zdb2test_0-rg \
-t SUNW.HASStoragePlus \
-p zpools=db2hapool \
-p FilesystemMountPoints=/global/opt/IBM \
zdb2test-has
```

10. Enable the DB2 resource group:

```
# clresourcegroup online -M db2_zdb2test_0-rg
```

The resource group `db2_zdb2test_0-rg` is online on `mdex4200-4`. The LOFS mount is automatically performed in the non-global zones as the `HAStoragePlus` resource goes online.

11. Mount the cluster file system on another node, e.g., `mdex4200-5`.

```
# mount -F lofs /global/opt/IBM /zones/zone1/root/global/opt/IBM
```

The manual LOFS mount for `/global/opt/IBM` on the other node allows validation of the DB2 HA resource in `zone1`. When the DB2 HA resource is created, the Solaris Cluster software validates the environment. Make sure the DB2 agent executables that are located in `/global/opt/IBM` are available whenever the DB2 resource group needs to fail over from `mdex4200-4:zone1` to `mdex4200-5:zone1`.

When the DB2 HA resource is created, the Solaris Cluster software validates the environment. Make sure the DB2 agent executables that are located in `/global/opt/IBM` are available whenever the DB2 resource group needs to fail over from one node in `zone1` to another.

Manual LOFS mount is used in this example as a proof of concept. The manual LOFS mount will not survive a node reboot, so in a production environment, use one of the following approaches:

- DB2 executables on local disk, failover, or cluster file systems
- DB2 agent executables on local disk, RTR file on local disk in `/usr/cluster/lib/rgm/rtreg`
- DB2 instance files on failover or cluster file systems

12. Log in to `zone1`, and set the DB2 global register variable and database manager configuration parameter for failover control:

```
# su - zdb2test -c "db2set DB2_NUM_FAILOVER_NODES=1"  
# su - zdb2test -c "db2 update dbm cfg using START_STOP_TIME 2"
```

13. Create the DB2 HA resource in `zone1`.

```
# clrs create -g db2_zdb2test_0-rg \  
-t IBM.db2udb_zdb2test \  
-p INSTHOME=/db2hapool/zdb2test \  
-p DB2INSTANCE=zdb2test \  
-p DB2_NODE_NUMBER=0 \  
-p Resource_dependencies=zdb2test-lh,zdb2test-has \  
db2_zdb2test_0-rs
```

14. Verify resource group and resources status.

```
# clresourcegroup show
Resource Group:                db2_zdb2test_0-rg
RG_description:                <NULL>
RG_mode:                       Failover
RG_state:                      Managed
Failback:                      False
Nodelist:                      mdex4200-4:zone1 mdex4200-5:zone1

--- Resources for Group db2_zdb2test_0-rg ---

Resource:                      zdb2test-lh
Type:                          SUNW.LogicalHostname:2
Type_version:                  2
Group:                         db2_zdb2test_0-rg
R_description:
Resource_project_name:        default
Enabled{mdex4200-4:zone1}:    False
Enabled{mdex4200-5:zone1}:    False
Monitored{mdex4200-4:zone1}: True
Monitored{mdex4200-5:zone1}: True

Resource:                      zdb2test-has
Type:                          SUNW.HAStoragePlus:4
Type_version:                  4
Group:                         db2_zdb2test_0-rg
R_description:
Resource_project_name:        default
Enabled{mdex4200-4:zone1}:    True
Enabled{mdex4200-5:zone1}:    True
Monitored{mdex4200-4:zone1}: True
Monitored{mdex4200-5:zone1}: True

Resource:                      db2_zdb2test_0-rs
Type:                          IBM.db2udb_zdb2test
Type_version:                  1.0
Group:                         db2_zdb2test_0-rg
R_description:
Resource_project_name:        default
Enabled{mdex4200-4:zone1}:    False
Enabled{mdex4200-5:zone1}:    False
Monitored{mdex4200-4:zone1}: True
Monitored{mdex4200-5:zone1}: True
```

At this point we have the DB2 V9.1 using ZFS and the LOFS mounted cluster file system in a non-global zone and can switch over.

Test Cluster Failover Within Solaris Containers

Bring online the resource group:

```
# clresource enable -g db2_zdb2test_0-rg +
```

Assume at this point that the resource group `db2_zdb2test_0-rg` is online at `mdex4200-5`. Then test the failover of the DB2 instance and associated resources from `zone1` on `mdex4200-5` to `zone1` on `mdex4200-4`.

```
# clresourcegroup switch -n mdex4200-4:zone1 db2-zdb2test_0-rg
```

For More Information

Here are some additional resources:

- Downloads for Solaris Cluster 3.2 software:
<http://www.sun.com/download/products.xml?id=4581ab9e>
- Sun training courses at <http://www.sun.com/training/>:
 - Sun Cluster 3.2 Administration (ES-345 and VC-ES-345)
 - Sun Cluster 3.2 Advanced Administration (ES-445)
 - Sun Certified System Administrator for Sun Cluster 3.2 Software (CX-310-345)
- Support:
 - Register your system with Sun Connection:
<http://www.sun.com/bigadmin/hubs/connection>
 - Sun Services: <http://sun.com/service>
 - SunSolve Online: <http://sunsolve.sun.com>
- Open source resources:
 - OpenSolaris™ community: <http://www.opensolaris.org/os/>
 - OpenSolaris community for HA Clusters:
<http://www.opensolaris.org/os/community/ha-clusters/>
- Developer forum for clustering: <http://forum.java.sun.com/forum.jspa?forumID=842>
- Documents available on <http://docs.sun.com>:
 - [*Sun Cluster 3.2 Software Collection for Solaris OS*](#)
 - [*Sun Cluster Quick Start Guide for the Solaris OS*](#)
 - [*Sun Cluster Software Installation Guide for Solaris OS*](#)
 - [*Sun Cluster 3.2 Data Services Collection for Solaris OS \(x86 Platform Edition\)*](#)
- Sun Cluster wiki: <http://wikis.sun.com/display/SunCluster/Home>
- Related sites and articles:
 - *IBM DB2 Universal Database and High Availability on Sun Cluster 3.X:*
<ftp://ftp.software.ibm.com/software/data/pubs/papers/suncluster.pdf>
 - *Deploying IBM DB2 UDB V8.2.x in Containers in the Solaris 10 OS:*
http://www.sun.com/bigadmin/features/articles/db2_containers.html
 - Solaris Cluster web site: <http://www.sun.com/software/solaris/cluster/index.xml>
 - Solaris Cluster hub on BigAdmin: <http://www.sun.com/bigadmin/hubs/sc/>
 - Database Collection on BigAdmin:
<http://www.sun.com/bigadmin/collections/database.html>
 - Solaris Cluster FAQ: <http://www.sun.com/software/solaris/cluster/faq.jsp>

- Events:
 - Worldwide developer events, including “Using Solaris 10 Containers and Zones”:
<http://developers.sun.com/events/>
 - Other current events: <http://www.sun.com/events/index.jsp>

Licensing Information

Unless otherwise specified, the use of this software is authorized pursuant to the terms of the license found at http://www.sun.com/bigadmin/common/berkeley_license.html.