OS Snapshots for Backup

Utilizing operating system snapshots for quick and painless Oracle database backup and restore.

By Kenny Gorman (kenny@kennygorman.com)

Toolbox: *This example utilizes Oracle 8.1.7.2.0 on SuSE Linux 7.3 with the 2.4.16-64GB_smp kernel. It assumes basic knowledge of Oracle Hot Backup methods, and the Linux operating system. The Oracle database must be running in archive log mode. It is also necessary to have root access to perform the examples shown.*

As the typical database size grows larger and larger, even the most ambitious backup plans begin to strain. With larger data volumes come longer backup times. In today's 24/7 world, allowing a database to be at risk and/or run at degraded performance due to a backup for long periods of time is just unacceptable. Many companies and institutions turn to incremental backups, or utilize expensive disk-mirroring technology to help keep the backup window time down. Some companies simply take fewer backups. Obviously, these techniques may require lots of expensive hardware or add complication and at worst, risk corporate data.

In addition, many database administrators take backups to disk in order to alleviate long backup windows while waiting for tape (and sometimes to reduce the Mean Time To Recover). Taking a backup to disk takes additional expensive disks, arrays, or NFS servers.

But there is another option that is under utilized by most database administrators. This technique is called OS file system snapshots. OS snapshots take a fraction of the time of file copies and are many times faster than backups directly to tape devices. OS snapshots are a simple and effective technique for reducing the backup window for large databases. OS snapshots can also reduce the amount of disk needed for backups to disk. In many cases it can be 10% of the total amount of disk.

Utilizing OS snapshots can be a useful technique to any database administrator and some system administrators running databases on UNIX and Linux.

Background / Overview

OS snapshots are a facility of the Linux LVM (Logical Volume Manager). A snapshot volume, once mounted, contains all the files that existed on the logical volume frozen at the point in time when the snapshot was taken. A snapshot volume can exist for one logical volume. You can have multiple snapshots of a single logical volume. In our case, the files on the snapshot volume are Oracle datafiles. Before we snapshot the logical volume where our Oracle datafiles reside, we will place Oracle in Hot Backup Mode. Once the snapshot is complete, we remove Oracle from Hot Backup Mode. Depending on the activity on your file system, the time for a snapshot to complete can take mere minutes. Once the snapshot is taken, then the backup to tape can occur from the snapshot, and Oracle remains unaffected and is in service. Snapshots are a facility of the volume manager software. Increasingly, UNIX vendors and third parties are offering volume managers that have snapshot facilities. In this article we will explore Linux LVM snapshots.

Kenny Gorman

Snapshots work for backups because Oracle has a facility called Hot Backup Mode. Hot Backup Mode allows for recovery of inconsistent (or "fuzzy") datafiles using the archive logs. When in Hot Backup Mode, Oracle writes not just the change vector for a block, but also the entire value for the change to the redo log. The Oracle datafiles are still written to, but that doesn't matter, because the archive logs hold all transactions that took place. When a snapshot takes a "picture" of the datafiles in Hot Backup Mode, they are inconsistent (or "fuzzy"). We don't care that the files are inconsistent because during recovery, Oracle will apply the archive logs and bring the database into a consistent state.

This article assumes your system is Linux 2.4.16 kernel with support for LVM and ReiserFS. The system used in this example is SuSE Linux 7.3, and comes with the needed software bundled right in. If you are unfamiliar with installation of software on Linux and rebuilding your kernel, I recommend you utilize the SuSE distributions. Of course because of the nature of Linux, pretty much any Linux distribution can be set up with the needed software. This article also assumes you are using Oracle 7.3+ and the database is running in archive log mode.

Step 1: Preliminary Setup

In order to create a snapshot, you must first have a logical volume to take a snapshot of. In order to create a logical volume, you must first have a disk to assign to the logical volume via a volume group. These steps are also outlined at www.sistina.com, the creator of the LVM software.

First the disk will need to be formatted using fdisk, and, in this case, a physical volume that spans the entire disk will

be created. It is important to note that the disk type needs to be assigned the type 8e. This is shown below.

```
$>su –
$>fdisk /dev/sdb

The number of cylinders for this disk is set to 4492.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): p

Disk /dev/sdb: 255 heads, 63 sectors, 4492 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-4492, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-4492, default 4492):
Using default value 4492

Command (m for help): t
Partition number (1-4): 1
Hex code (type L to list codes): 8e
Changed system type of partition 1 to 8e (Linux LVM)

Command (m for help): p

Disk /dev/sdb: 255 heads, 63 sectors, 4492 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/sdb1             1      4492  36081958+  8e  Linux LVM

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: If you have created or modified any DOS 6.x
partitions, please see the fdisk manual page for additional
information.
Syncing disks.
$>
```

Now I have a physical volume to work with. To create the volume group and subsequent logical volume, we must first initialize the physical volume.

```
$>pvcreate /dev/sdb1
pvcreate – physical volume "/dev/sdb1" successfully created
```

This creates a volume group descriptor area (VGDA) at the start of the disks. Next, the disks need to then become part of a disk group. In this case, I define one disk group for the physical partition.

```
$>vgcreate datagrp /dev/sdb1
vgcreate – INFO: using default physical extent size 4 MB
vgcreate – INFO: maximum logical volume size is 255.99 Gigabyte
vgcreate – doing automatic backup of volume group "datagrp"
vgcreate – volume group "datagrp" successfully created and
activated
```

This creates a disk group assigned to a physical partition. Now I need to assign the volume group to a logical volume. I will create the logical volume and leave some extra space for my snapshot. This extra space is arbitrary and depends on your transaction volume and length of snapshot for space consumption. In my case, I make it very large (relatively) so there is no problem with space. I create the volume as 32GB, thus leaving approximately 4GB for the snapshot (36GB disk—32GB volume).

```
$>lvcreate –L32G –ndatavol datagrp
lvcreate – doing automatic backup of "datagrp"
lvcreate – logical volume "/dev/datagrp/datavol" successfully
created
```

Once we have the logical volume then we need to create a filesystem on top of it. For this example, I will use ReiserFS. Details on ReiserFS can be found at www.reiserfs.com. You should carefully evaluate the requirements of your system to be sure ReiserFS is the correct choice.

```
$>mkreiserfs /dev/datagrp/datavol
Initializing journal - 0%....20%....40%....60%....80%....100%
Syncing..ok
```

This may take some time to create depending on your system. Once this is done you can mount it. I have to create a directory to mount it to in normal UNIX fashion.

```
$>mkdir /oradata
$>chown oracle:dba /oradata
$>mount /dev/datagrp/datavol /oradata
```

Now that the volume is mounted, then you can go ahead and create a database if you want. For this example, I am putting all the datafiles on one volume. In your implementation that probably won't make sense, but this should give you a good idea of how to use snapshots nonetheless.

Creating a snapshot volume

You have seen how to create a regular logical volume above. Now we need to create the snapshot. Snapshots are simply logical volumes created with special syntax. The snapshot volume syntax specifies the volume you are taking the snapshot of.

First we want to create a mount point for the snapshot.

```
$>mkdir /oradata_snap
$>chown oracle:dba /oradata_snap
```

Before you take the snapshot, you want to be sure to place the tablespace(s) that have datafiles on your logical volume in Hot Backup Mode. Otherwise, your backup will be corrupted. In our case, the entire database is on one volume. So it makes sense to put all the tablespaces in Hot Backup Mode all at once. You probably won't want to put all tablespaces in Hot Backup Mode on your production systems. Instead, you would cycle through your tablespaces, putting them in Hot Backup Mode one by one, taking snapshots, and then removing them from Hot Backup Mode.

Put all the tablespaces in Hot Backup Mode.

```
SQL> DECLARE
  2   sqlstmt varchar2(100);
  3   CURSOR tab_cur IS
  4   SELECT tablespace_name FROM dba_tablespaces;
  5  BEGIN
  6   FOR tab_rec IN tab_cur
  7    LOOP
  8     sqlstmt := 'ALTER TABLESPACE '||tab_rec||' BEGIN BACKUP';
  9     EXECUTE IMMEDIATE sqlstmt;
 10   END LOOP;
 11  END;
12 /
SQL> PL/SQL procedure successfully completed.
```

Once the tablespaces are in Hot Backup Mode, we can take the snapshot of the volume.

```
$>lvcreate –L2G –s –noradata_backup /dev/datagrp/datavol
lvcreate – WARNING: the snapshot will be automatically disabled
once it gets full
lvcreate – INFO: using default snapshot chunk size of 64 KB for
"/dev/datagrp/oradata_backup"
lvcreate – doing automatic backup of "datagrp"
lvcreate – logical volume "/dev/datagrp/oradata_backup"
successfully created

$>mount /dev/datagrp/oradata_backup /oradata_snap
mount: block device /dev/datagrp/oradata_backup is write-
protected, mounting read-only
```

You have created the snapshot! You can now remove Oracle from Hot Backup Mode.

```
SQL> DECLARE
  2    sqlstmt varchar2(100);
  3    CURSOR tab_cur IS
  4    SELECT tablespace_name FROM dba_tablespaces;
  5   BEGIN
  6    FOR tab_rec IN tab_cur
  7     LOOP
  8      sqlstmt := 'ALTER TABLESPACE '||tab_rec||' END BACKUP';
  9      EXECUTE IMMEDIATE sqlstmt;
 10     END LOOP;
 11   END;
13 /
SQL> PL/SQL procedure successfully completed.
```

You can now copy (tar, cpio, Netbackup, etc.) your datafiles to tape from the snapshot at your leisure without needing Oracle to be in Hot Backup Mode, or worse, taking Oracle down. You will also need to be sure to back up the archive logs for the period that the database was in Hot Backup Mode.

```
$>tar –cvf /dev/rmt0 /oradata_snap/* /oraarch/*
```

Once the backup is complete, unmount and remove the volume. Once you do this, the snapshot is lost forever. Snapshots don't keep the data between reboots, so you will not want to rely on the snapshot itself for a backup, you will want to copy the data off to a tape drive or some other type of permanent media.

```
$>umount /oradata_snap
$>lvremove –f /dev/snapgrp/oradata_backup
lvremove – doing automatic backup of volume group "datagrp"
lvremove – logical volume "/dev/datagrp/oradata_backup"
```

Restoration

Of course, none of this is any good if you don't test the restore. In order to restore, the tape media or other storage would need to be utilized, and the copy of the data redirected back to the original source directory. For instance, using the above example, I backed up my snapshot from /oradata_snap and archive logs from /oraarch. I would redirect the restore to put the files from /oradata_snap on /oradata, and the files from /oraarch back to the /oraarch volume. At this point, the database could be recovered and all would be well.

```
$>tar –xvf /dev/rmt0 /oradata

SQL>…
SQL>Recover database until cancel.
SQL>…
```

Weaving snapshots into your backup scheme

Obviously, there is more to using snapshots when you consider most production environments. Mainly you will want to script both the snapshots and the Hot Backup Mode. You will also need to consider archive logs, and make sure you have the correct set of archive logs to recover the database. In my environments, I use the following order of events:

1. switch archive logs
2. start backup for tablespace 1
3. snapshot filesystem for tablespace 1
4. end backup for tablespace 1
5. repeat for all tablespaces
6. create control file to trace
7. switch archive logs
8. snapshot the archive log filesystem
9. run tape backup software to 'grab' all of the snapshot volumes

Monitoring the completion and ensuring a successful snapshot are, of course, essential. I recommend reading the documentation at www.sistina.com to understand all of the different parts of the LVM. There are lots of powerful things you can do with it. You may over time need to extend the sizes of the data and/or snapshot volumes. Management of the sizes and interdependencies of the data and its snapshot are important.

As of this writing, LVM was still in beta. So care should be taken in how you or if you deploy it in a production environment. I recommend keeping up to date on the latest version and subscribing to the LVM mailing list at: mailto:linux-lvm@sistina.com.

Conclusion

Snapshots can be really cool, and can reduce your backup window substantially. They can also save on disk space by alleviating the need to have extra disks just for backups. Additionally, the components shown in this article are free. They come with some major Linux distributions, but can also be downloaded and compiled separately.

Kenny Gorman (kenny@kennygorman.com) is an independent Oracle consultant. He provides Oracle DBA services to clients around the San Francisco Bay Area. He specializes in High Availability, Monitoring and Management, and Veritas. He is an OCP and Oracle Master and has over six years of experience as an Oracle DBA at various companies and startups in and around the Bay Area.